invensys®

WorkPlace Tech Tool 4.0
Engineering Guide

I/A Series®
Software for Intelligent Buildings

Printed in U.S.A. 12/03 F-27254

Copyright Notice

The confidential information contained in this document is provided solely for use by Invensys Building Systems employees, licensees, and system owners, and is not to be released to, or reproduced for, anyone else. Neither is it to be used for reproduction of this control system or any of its components.

All specifications are nominal and may change as design improvements occur. Invensys Building Systems shall not be liable for damages resulting from misapplication or misuse of its products.

Invensys Building Systems

1354 Clifford Avenue (Zip 61111)
P.O. Box 2940
Loves Park, IL 61132-2940
United States of America
www.invensysibs.com

© 2003 Invensys Building Systems
All rights reserved. No part of this document may be photocopied or reproduced by any means, or translated to another language without prior written consent of Invensys plc.

Invensys, I/A Series, and NETWORK 8000 are trademarks of Invensys plc and its subsidiaries and affiliates.
Adobe and Acrobat are trademarks of Adobe Systems Incorporated.
Echelon, LON, LONMAKER, LONMARK, LONTALK, and LONWORKS are trademarks of Echelon Corporation.
ICELAN 2000 is a trademark of IEC Intelligent Technologies.
Microsoft, Windows, Windows NT, and Visio are trademarks of Microsoft Corporation.

Table of Contents

Preface xix

Purpose of this Manual xix
Applicable Documentation xx
Conventions Used in this Manual xxi
Acrobat (PDF) Conventions xxi
Abbreviations Used in this Manual xxii
Manual Summary xxiii

Chapter 1 **Introduction 1**

I/A Series Hardware Products 2
MicroNet Standard Controllers 2
Common Controller Features 2
Model Number Code 3
Universal Inputs 3
MN 50, 100,150, 200 Hardware Platforms 3
MN 110, 130
Hardware Platform 5
VAV Hardware Platforms 6
Controller Firmware Revisions 7
Profile Version Numbers 8
MicroNet MN 800 Controller 9
MicroNet Digital Wall Sensors 10
Common Sensor Features 10
Sensor Models and Functions 11
Diagnostic Functions of LCD Screen Models 12
WorkPlace Communications Adapters 13
Common Adapter Features 13
I/A Series Software Products 14
The WorkPlace Tech Tool 14
Application Creation and Modification 15
Online Diagnostics 15
Project Based Folders 16
MicroNet VAV Flow Balance Software 16
Third-Party LonWorks Products 17

Network Management Tools 17

Chapter 2 **Object Programming Basics** **19**

Objects in an I/A Series MicroNet Controller 20
An Object as an Algorithm 21
Object Inputs and Outputs 22
Analog and Digital Inputs and Outputs 22
Physical Address Inputs and Outputs 22
Linked with Other Control Objects 23
One Output to Many Inputs 24
Configuration Properties 24
Data (Number) System 25
Analog Data 25
Digital Data 25
Not Active (NA) 26
External Data Exchange in a MicroNet Controller 26
Use of Controller Object Memory 26
Objects in WP Tech 27
Shape Stencils 27
WP Tech Stencils 29
Creating New (Custom) Stencils 30
Control Objects 30
Tags 31
Linking Objects (and Tags) 33
Using Variable Definition and Reference Tags 37
Engineering Process Overview 41
Custom Applications 41
Pre-engineered Applications 42
Downloading Applications 42
Workstation Addressing Wizard 42
Realtime Application Checkout (Diagnostics) 44
Control Object Categories 44
I/O Point Objects 44
Input Point Objects 45
Output Point Objects 45
I/O Differences Among Hardware Platforms 46
Functional Objects 46
Alarm Objects 46
Logic and Math Control Objects 47
Loop and Process Control Objects 47
Timer and Sequence Control Objects 48
Schedule Control Objects (MN 800) 48
Migrating WP Tech 2.0 or 3.0 Projects into WP Tech 3.2 48
Opening WP Tech 3.1 Projects in WP Tech 3.2 49
Migrating Projects into WP Tech 4.0 49
Migrating from WP Tech 2.0, 3.0, or 3.1 49
Migrating WP Tech 3.2 Projects into WP Tech 4.0 49

Chapter 3 **Understanding Programming Boundaries** 51

Resource Tags In General 52
Resource Tag Rules 53
Controller Type Considerations 53
Controller I/O Points 54
Controller Inputs 54
Controller Outputs 54
Integral I/O Points 54
I/O Point Capacities (by Controller Model) 55
WP Tech Representations 55
MicroNet Sensor Attributes 56
WP Tech Representations 56
S-Link Sensor (Sensor Tags) 58
Sensor Tags 59
Applying the Sensor Tags (S-Link Object) 65
The Most Basic Sensor Tags 66
Occupancy Override Sensor Tags 67
Setpoint Sensor Tags 68
Fan1 and Fan2 Sensor Tags 69
HVAC Mode1 and Mode 2 Sensor Tags 70
Fan and Mode Tags Example 71
Emergency Heat Sensor Tags 71
LCD Screen Sensor Tags 72
MicroNet Sensor Configuration Parameters Not in Sensor Tags 74
Other Resource Tags (Standard Controllers) 74
Schedule Tags 75
Schedule Tags 76
Applying the Schedule Tags 76
Controller Clock 77
Clock Initialization 77
Controller Schedule 77
Schedule Outputs 78
Schedule Override 79
Time Synchronization 79
Controller Reset Considerations 79
Schedule Tag Example 80
Clock Tags (MN 800) 80
Controller Memory (RAM and EEPROM) 80
Standard Controllers 81
MN 800 Controllers 81
Drawing Information Storage 81
WP Tech Monitoring 82
Custom Objects 82
Logical Boundary 84
Creating Custom Objects 84
Reusing Custom Objects 85
Custom object 86

Chapter 4 **Anatomy of a Control Object** **87**

Object Properties 88
Configuration Properties 88
Name and Description 89
Process Time 89
Algorithm-related Configuration Properties 91
Input Properties (inputs) 92
Physical Address Inputs 92
Input Data Classes 92
Input Data Sources 93
Output Properties 96
Physical Address Outputs 96
Output Data Classes 97
Common Object Behavior 99
Mixing of Data Types 99
Inverted Inputs 101
Digital Class Inputs Inverted 101
Analog Class Inputs Inverted 101
Prioritized Inputs 102

Chapter 5 **Control Objects** **103**

Objects Grouped by Stencils 104
Objects Grouped Alphabetically 105
Other Objects on Stencils 106
Objects Not On Stencils 106
Abs Sub / Div 108
Properties 108
Applying the Object 109
Add / Add 110
Properties 110
Applying the Object 111
Add / Div 112
Properties 112
Applying the Object 113
Analog Alarm 114
Properties 114
Applying the Object 116
High Alarm Sequence 116
Low Alarm Sequence 116
High and Low Alarm Activation 117
Return from High and Low Alarm 119
Example Applications 121
Analog Input 123
Properties 123
Applying the Object 125
Thermistor / Balco /

Platinum RTD Sensors 126
Milliamps / Volts 126
Resistance (1kW and 10kW) 132
Offset Calibration 133
Filter 134
Analog Output 135
Properties 135
Applying the Object 136
Current Driven Device Example 137
Voltage Driven Device Example 138
Analog Output Priority 139
Properties 139
Applying the Object 141
Priority Inputs and Values 142
Current Driven Device Example 143
Voltage Driven Device Example 144
AND / AND 145
Properties 145
Applying the Object 146
Example Application 148
AND / OR 149
Properties 149
Applying the Object 150
Average 152
Properties 152
Applying the Object 153
Binary Alarm 154
Properties 154
Applying the Object 156
Alarm Sequence and Alarm Activation 156
Binary Alarm 157
Return from Binary Alarm Sequence and Activation 157
Example Applications 159
Binary Encoder 161
Properties 161
Applying the Object 162
Binary Encoded Output 162
Example Application 164
Binary Input 165
Properties 165
Applying the Object 167
Direct 168
Reverse 169
Pulse 169
Binary Output 173
Properties 173
Applying the Object 174

Action 174
Direct 175
Reverse 175
Calendar 177
Properties 177
Applying the Object 181
Program Type 181
Event/Occurrence Required Properties 183
Leap Year Operation 183
Examples 184
Clocked SR 186
Properties 186
Applying the Object 187
Compare 189
Properties 189
Applying the Object 190
Example Application 191
Compare 2 192
Properties 192
Applying the Object 194
Reset and Not Active 194
Control Override 195
Properties 195
Applying the Object 196
Example Applications 197
Count Down 198
Properties 198
Applying the Object 199
Count Up 200
Properties 200
Applying the Object 201
COV Priority 202
Properties 203
Applying the Object 204
Operation Sequence 205
Example 205
Curve Fit 207
Properties 207
Applying the Object 209
Examples 209
Example 1 - Valve Characterization 210
Example 2 - Curve Fit Object Cascade 212
Example 3 - Sensor Characterization 213
Demux Select 215
Properties 215
Applying the Object 217
Example 217

Dual Delay 218
Properties 218
Applying the Object 219
Dual Minimum 221
Properties 221
Applying the Object 222
Example 223
DUI Expander 224
Properties 225
Applying the Object 226
Action Configuration 227
Status Flags Output 228
Input Resistance Combinations 228
Example 231
Enthalpy 232
Properties 232
Applying the Object 236
Configuration 236
Inputs 237
Outputs 238
Input / Output Units 240
Standard Atmospheric Data for Altitudes 240
Conversions 241
Example 241
Event Indicator 243
Properties 243
Applying the Object 244
Example 245
EXOR 246
Properties 246
Applying the Object 247
Fan Speed 248
Properties 248
Applying the Object 250
Number of Speeds Property 251
Enable / Disable 251
Digital Fan Speed Control
(Single, Two, or Three Speed) 252
Analog Fan Control (Variable Speed) 253
Examples 253
Filter 257
Properties 257
Applying the Object 258
How the Filter Algorithm Works 259
Example Application 259
Floating Actuator 260
Properties 260
Applying the Object 263

Without Feedback (Modes 0 and 2) 264
With Feedback (Modes 1 and 3) 265
Examples. 268
Floating Actuator Priority 270
Properties 271
Applying the Object 273
Priority Inputs and Values 274
Without Feedback (Modes 0 and 2) 275
With Feedback (Modes 1 and 3) 277
Examples. 279
High Select 281
Properties 281
Applying the Object 282
Interlock 283
Properties 283
Applying the Object 284
Interlock Sequences Explained 285
Timing Diagrams 286
Example 287
Interstage Delay (3) 288
Properties 288
Applying the Object 290
Interstage Sequence 290
Example 292
Interstage Delay (6) 293
Properties 293
Applying the Object 295
Interstage Sequence 295
Example 297
Interstage Delay (10) 298
Properties 298
Applying the Object 300
Interstage Sequence 301
Example 302
Latch 303
Properties 303
Applying the Object 304
Digital Latch 305
Sample and Hold 306
Limit 307
Properties 307
Applying the Object 308
Limit Thermostat 309
Properties 309
Applying the Object 311
Example 311

Loop Sequenced 312
Properties 312
Applying the Object 315
General Rules 316
Ramp Start 316
Loop Differences 317
Cooling (Loop1) 317
Proportional Control 317
Integral Control 318
Derivative Control 318
Ramp Function in Cooling Demand 318
Heating (Loop2) 319
Proportional control 319
Integral control 319
Derivative Control 320
Ramp Function in Heating Demand 320
Economizer (Loop3) 321
Controlled 321
Auto Range 322
Ramp Function Economizer Demand 324
Economizer Rules 324
General Guidelines for Setting Up Loop Control 325
Example Application 326
Loop Single 328
Properties 328
Applying the Object 329
Proportional only (P) Control 330
Proportional plus Integral (PI) Control 330
Proportional plus Integral plus Derivative (PID) Control 331
Action 331
Ramp Start Function 331
Example 332
General Guidelines for Setting Up Loop Control 333
Low Select 334
Properties 334
Applying the Object 335
MA Volume 336
Properties 336
Applying the Object 337
Example Application 338
Minimum Off 339
Properties 339
Applying the Object 340
Example 341
Minimum On 342
Properties 342
Applying the Object 343
Momentary Start / Stop 345
Properties 345

Applying the Object 346
Start Pulse and Stop Pulse 347
Pulse in Progress 347
Momentary Enable 347
Mul / Add 348
Properties 348
Applying the Object 349
Mul / Div 350
Properties 350
Applying the Object 351
Off Delay 352
Properties 352
Applying the Object 353
On Delay 355
Properties 355
Applying the Object 356
OR / AND 358
Properties 358
Applying the Object 359
OR / OR 361
Properties 361
Applying the Object 362
OSS 364
Properties 364
Applying the Object 376
Optimum Start and Optimum Stop Functions 376
Optimum Start / Stop Modes 376
Input Select / Input Format 377
Zone Mode 379
Outside Air Mode 387
Application Examples 402
Pressure Transducer 406
Properties 406
Applying the Object 408
Calibration Values 408
Object Outputs 408
Example 409
Priority Input (2) 410
Properties 410
Applying the Object 411
Example 411
Priority Input (4) 412
Properties 412
Applying the Object 414
Example 414
Priority Value Select 415
Properties 415

Applying the Object 416
Priority Type Objects Compared 417
Priority Value Select Example 417
PWM 418
Properties 418
Applying the Object 419
General Behavior 420
Applications and Examples 421
Time-Proportioned Control 421
Fixed Duty Cycle Control 425
Compensated Duty Cycle Control 426
PWM Priority 427
Properties 427
Applying the Object 429
General Behavior 429
Priority Inputs and Values 430
Applications and Examples 431
Time-Proportioned Control 431
Fixed Duty Cycle Control 435
Compensated Duty Cycle Control 435
Ramp 437
Properties 437
Applying the Object 439
Standard Analog Ramp 439
Example Analog Ramp 440
Step Change Ramp 441
Example Step Change Ramp 442
Reset 443
Properties 443
Applying the Object 444
Direct Reset Example 445
Reverse Reset Example 446
Reset Ratio 446
Control Logic Example 447
Schedule 7-Day 449
Adding a Schedule 7-Day Object to a Drawing 450
Properties 451
Applying the Object 456
Schedule Enable/Disable 456
Forced Override 456
Valid/Invalid Time 456
Schedules 457
Outputs 460
Example 462
Select 463
Properties 463
Applying the Object 464
Control Logic Example 465

Sensor Input 466
Properties 466
Applying the Object 469
Input Sensor Type 469
Switch Type 470
Filter Constant 472
Offset Calibration 472
Status Output 473
Sequence (3) 474
Properties 474
Applying the Object 477
Linear Sequence Mode 477
Analog Sequence Mode 478
Vernier Sequence Mode 479
Sequence (6) 481
Properties 482
Applying the Object 483
Linear Sequence Mode 483
Analog Sequence Mode 485
Vernier Sequence Mode 486
Sequence (10) 488
Properties 488
Applying the Object 490
Linear Sequence Mode 491
Analog Sequence Mode 492
Vernier Sequence Mode 493
Setpoint Control 496
Properties 496
Applying the Object 497
Single Setpoint Mode vs. Dual Setpoint Mode 498
Single Setpoint Mode 498
Dual Setpoint Mode 499
Example 500
SqRt Mul / Add 501
Properties 501
Applying the Object 502
Example Application 503
SR Flip-Flop 504
Properties 504
Applying the Object 505
Example 506
Step Driver 507
Properties 507
Applying the Object 509
Reset Operation 510
Examples 510
Sub / Add 511
Properties 511

Applying the Object 512
Sub / Div 513
Properties 513
Applying the Object 514
Sub / Mul 515
Properties 515
Applying the Object 516
Sub / Sub 517
Properties 517
Applying the Object 518
Thermostat 519
Properties 519
Applying the Object 520
Direct Acting and Reverse Acting 520
Example 521
Thermostat 2 522
Properties 522
Applying the Object 523
Direct Acting and Reverse Acting 524
Example 525
VAV Actuator 526
Properties 526
Applying the Object 528
Analog Value Inputs 529
Digital Value Inputs 529
Normal Operation 530
Examples 531

Chapter 6 **LonWorks Network Data Exchange 533**

LonWorks Network Architecture 534
WP Tech on an I/A Series LonWorks Network 534
Data Exchange 534
SNVTs 536
SNVT Characteristics 536
SNVT Types 536
SNVT Formats 536
NV Implementation in I/A Series MicroNet Controllers 537
MicroNet Standard Controllers 538
Invensys LonMark Profiles 538
Profile Function 538
LonMark Compliancy 539
LonMark Profile Representations 539
WP Tech Representations 542
Input Profile Tags (NCIs and NVIs) 543
Output Profile Tags (NVOs) 545
Standard Controllers External Interface File (XIF) 547
MicroNet MN 800 Controllers 548

User-Definable Profile 548
SNVT Objects 548
NCI Objects (nciType) 548
NVI Objects (nviType) 549
NVO Objects (nvoType) 549
Mandatory (Default) SNVT Objects 550
Adding SNVT Objects 550
SNVT Object Name and Index Number 551
MN 800 External Interface File (XIF) 552
NCI objects 553
Applying an NCI Object 553
Adding an NCI Object 554
Naming an NCI Object 554
Setting Properties of an NCI Object 554
Conversion Selection 554
Properties 556
Mandatory (Default) NCI Objects 558
NCI Object on Network Variables Stencil 563
Engineering Units 563
Available SNVTs 564
NVI objects 565
Applying an NVI Object 565
Adding an NVI Object 565
Naming an NVI Object 566
Setting Properties of an NVI Object 566
Conversion Selection 566
Properties 567
Mandatory (Default) NVI Object 570
NVI Object on Network Variables Stencil 570
Advanced Level Inputs and Outputs 571
Engineering Units 572
Available SNVTs 573
NVO objects 574
Applying an NVO Object 574
Adding an NVO Object 574
Naming an NVO Object 575
Setting Properties of an NVO Object 575
Conversion Selection 575
Properties 576
Mandatory (Default) NVO Objects 579
NVO Object on Network Variables Stencil 580
Advanced Level Inputs 580
Available SNVTs 584
Unit Conversion 585
Network Management 590
Network Address 590
Network Variable Binding 590
Network Variable Updates 591
Send Heartbeat 591

Receive Heartbeat 591
Minimum Output Time 591
Minimum Propagation Time (MN 800) 592
Time Selection Guidelines 592
Configuration Properties 592
Other Considerations 593
WP Tech (Download, Upload, Monitor, Browse) and Device Addresses 593
WP Tech Device Address Assignments 595
WP Tech Real-time Data Monitoring and Point History Operation 595
Profile and Bindings Issues 595
LonWorks Message Services 596
MicroNet Controller Message Service Defaults 597
Other NV Data Exchange 597
LNC-100 and NV Polling 597
Point History Log 599
Tracked Data 599
Data Tracking Pens 599
Data Format 600
Application 600
Properties 601
Accessing Collected Data 601
Service Pin Operation 602
Diagnostic Indicators 604
MicroNet Controller LEDs 604
MN 50, 100, 110, 130, 150, 200, and VAV Series 604
MN 800 Series 606
S-Link Sensor Display 607
To Access Error and Alarm Indications 607

Appendix A **Memory Requirements** **609**

Memory Types 609
Controller Capacities 610
Standard Controller Memory 610
MN 800 Memory 611
Object Memory Requirements 612
WP Tech Statistics Function 612
Memory Requirements Per Object 613

Appendix B **Profiles and Network Data** **615**

Invensys LonMark Profiles 616
Profile by Model Number 616
Profile Versions 617
Fan Coil Profile 618
Fan Coil Profile Quick Reference 619
Heat Pump Profile 623
Heat Pump Profile Quick Reference 624
Roof Top Unit Profile 627

Roof Top Unit Profile Quick Reference 628
Satellite 1 Profile 631
Satellite 1 Profile Quick Reference 632
Satellite 2 Profile 637
Satellite 2 Profile Quick Reference 638
Satellite 3 Profile 643
Satellite 3 Profile Quick Reference 644
Satellite 4 Profile 649
Satellite 4 Profile Quick Reference 650
VAV Controller Profile 655
VAV Controller Profile Quick Reference 656
MN 800 User-Definable Profile 660
MN 800 Profile Quick Reference 661
SNVTs 662
SNVT to MicroNet Control Logic Data Exchange 662
Invalid data and not active (NA) 662
MN Controllers SNVTs Quick Reference 663
Invensys-defined Network Variable Types 696
Index **699**

Preface

Purpose of this Manual

This I/A Series™ WorkPlace Tech Tool Engineering Guide is a reference for programming I/A Series MicroNet™ standard controllers (MN 50, 100, 110,130, 150, 200, and VAV series) and the I/A Series MicroNet MN 800 controller, using the WorkPlace Tech Tool (WP Tech), Version 4.0. It provides a detailed description for each of the 80-plus types of control objects. Throughout this reference, control objects and related entities are depicted using the graphical shapes found in WP Tech. Also provided are explanations on how these MicroNet controllers store objects, process data, and operate with MicroNet sensors. Reference information on each of the Invensys LONMARK® profiles and available LONWORKS® network variables is also included.

WP Tech 4.0 is a PC-based program, designed for use with Windows® 2000 Professional or Windows XP and Visio® 2002. It is not designed for use with any other operating system, including Windows 98 or Windows NT®.

Procedures for using WP Tech are included in a companion manual, the *I/A Series WorkPlace Tech Tool 4.0 User's Guide*, F-27255.

It is assumed that readers of this manual understand basic HVAC concepts. An understanding of LONWORKS™ networking and communications is helpful. This manual is written for:

- Application engineers.
- Users who change hardware or control logic.
- HVAC technicians and field engineers.
- Service personnel who maintain I/A Series systems

Applicable Documentation

F-Number	Description	Audience	Purpose
F-27255	I/A Series WorkPlace Tech Tool 4.0 User's Guide	– Application Engineers – Installers – Start-up Technicians – Service Personnel	Provides step-by-step instructions for using the WorkPlace Tech Tool, version 4.0.
F-27278	I/A Series WorkPlace Tech Tool 4.0 Installation Instructions	– Application Engineers – Installers – Service Personnel – Start-up Technicians	Provides step-by-step instructions for installing WorkPlace Tech Tool, version 4.0.
F-27316	I/A Series WorkPlace Tech Tool 4.0 Release Notes	– Application Engineers – Installers – Service Personnel – Start-up Technicians	Provides important information on issues related to WorkPlace Tech Tool, version 4.0, that became apparent at release and are not fully documented elsewhere.
F-27317	I/A Series WorkPlace Tech Tool 4.0 Unlock Request Form	– Application Engineers – Installers – Service Personnel – Start-up Technicians	A form for requesting the unlock code for a WorkPlace Tech Tool, version 4.0, installation.
F-27318	I/A Series WorkPlace Tech Tool 4.0 Unlock Request Form for the UK	– Application Engineers – Installers – Service Personnel – Start-up Technicians	A form for requesting the unlock code for a UK installation of WorkPlace Tech Tool, version 4.0.
F-27319	I/A Series WorkPlace Tech Tool 4.0 Specification Data	– Application Engineers – Sales Personnel	Describes features and specifications of WorkPlace Tech Tool, version 4.0.
F-26617	I/A Series MicroNet MN 50 Series Controllers Installation Instructions	– Application Engineers – Installers – Service Personnel – Start-up Technicians	Provides step-by-step mounting and installation instructions for the I/A Series MN 50 Series Controllers. Also includes checkout and LED indication sections.
F-26887	I/A Series MicroNet MNL-11RF2 and MNL-13RF2 Controller Installation Instructions	– Application Engineers – Installers – Service Personnel – Start-up Technicians	Provides step-by-step mounting and installation instructions for the I/A Series MNL-11RF2 and MNL-13RF2 Controllers. Also includes checkout and LED indication sections.
F-26266	I/A Series MicroNet MN 100, 150, and 200 Series Controllers Installation Instructions	– Application Engineers – Installers – Service Personnel – Start-up Technicians	Provides step-by-step mounting and installation instructions for the I/A Series MN 100, 150, and 200 Series Controllers. Also includes checkout and LED indication sections.
F-26282	I/A Series MicroNet VAV Series (MNL-V1RVx and MNL-V2RVx) Controllers Installation Instructions	– Application Engineers – Installers – Service Personnel – Start-up Technicians	Provides step-by-step mounting and installation instructions for the I/A Series MNL-V1RVx and MNL-V2RVx VAV Controllers. Also includes checkout and LED indication sections.
F-26724	I/A Series MicroNet MN 800 Controller Installation Instructions	– Application Engineers – Installers – Service Personnel – Start-up Technicians	Provides step-by-step mounting and installation instructions for the I/A Series MN 800 Series Controller. Also includes checkout and LED indication sections.

F-Number	Description	Audience	Purpose
F-26284	I/A Series MicroNet VAV Series (MNL-V3RVx) Controller Installation Instructions	– Application Engineers – Installers – Service Personnel – Start-up Technicians	Provides step-by-step mounting and installation instructions for the I/A Series MNL-V3RVx VAV Controller. Also includes checkout and LED indication sections.
F-26277	I/A Series MicroNet MN-SX Series Sensors General Instructions	– Application Engineers – Installers – Service Personnel – Start-up Technicians	Provides step-by-step installation and checkout procedures for I/A Series MicroNet MN-SX Series Sensors. Also contains instructions for sensor operation.
F-26421	I/A Series MicroNet VAV Flow Balance User's Manual	– Application Engineers – Installers – Start-up Technicians – Service Personnel	Provides step-by-step instructions for using the MicroNet VAV Air Balance Software.
F-26987	I/A Series WorkPlace Tech Tool (version 3.2) User's Guide	– Application Engineers – Installers – Start-up Technicians – Service Personnel	Provides step-by-step instructions for using WorkPlace Tech Tool, version 3.2
F-26988	I/A Series WorkPlace Tech Tool (version 3.2) Engineering Guide	– Application Engineers – Service Personnel	Provides a reference for using WorkPlace Tech Tool, version 3.2 to program MicroNet controllers. Gives detailed descriptions for each of the Control Objects used with MN controllers.

Conventions Used in this Manual

These few conventions apply to this printed manual:

- Control objects types are always Capitalized, such as:

  The Analog Input object is an I/O point object.
- Menu commands appear in **bold**.
  Example — On the Special menu, point to **Security**, then click **Log On**.
- Italics is used for *emphasis* in a statement, such as:

  An Analog Input object has a physical address *input* used to specify which of the controller's universal input (UI) terminals that it monitors.

  It is also used when referring to a document, such as:

  Refer to the *WorkPlace Tech Tool 4.0 User's Guide*, F-27255.

Acrobat (PDF) Conventions

If you are reading this manual online in Adobe Acrobat™ (.PDF file format), numerous hypertext links exist, both in normal black text and in blue text.

- Hypertext links in this document include all entries in the Table of Contents and the Index, as well as cross-references within the body text. For ease of recognition, cross-reference links within the body text appear in blue type, for example Manual Summary. A link is indicated whenever the mouse pointer changes to a hand with a pointing finger.
- When viewing this guide with Adobe Acrobat, you can display various "bookmark" links on the left side of your screen by choosing "Bookmarks and Page" from the "View" menu. As with the links described above, these "bookmark" links will also cause the mouse pointer to change to a hand with a pointing finger.

Abbreviations Used in this Manual

AHU	Air Handling Unit
AO	Analog Output
DI	Digital Input
DO	Digital Output
EEPROM	Electrically Erasable Programmable Read-Only Memory
EPROM	Erasable Programmable Read-Only Memory
HVAC	Heating, Ventilating, and Air Conditioning
I/A	Intelligent Automation
I/O	Input / Output
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LON	Local Operating Network
FCS	Fan Coil Sensor
FTT	Free Topology Transceiver
LNMT	LONWORKS Network Management Tool
mA	milliAmperes
MicroNet (Controllers and Sensors)	I/A Series MicroNet Controllers and Sensors (Throughout this manual, these products are not to be confused with MicroNet 2000 Controllers and Sensors.)
MN Sensor	MicroNet Sensor
NCI	Network Configuration Input
NVI	Network Variable Input
NVO	Network Variable Output
OTP	One-Time-Programmable
PC	Personal Computer
PDF	Portable Document Format
PCMCIA	Personal Computer Memory Card International Association
PWM	Pulse Width Modulation
R2R	Resistor-to-Resistor
RAM	Random Access Memory
ROM	Read Only Memory
RTU	Roof Top Unit
S-LK	Sensor Link or S-Link
SNVT	Standard Network Variable Type
TO	Triac Output
UI	Universal Input
Vac, Vdc	Volts (Alternating Current or Direct Current)
VAV	Variable Air Volume
WP Tech	WorkPlace Tech Tool
XIF	External Interface File

Manual Summary

The I/A Series WorkPlace Tech Tool Engineering Guide contains six chapters, two appendices, and an index.

Chapter 1, **Introduction**, provides a brief overview of the various I/A Series hardware and software products, as well as a discussion of third-party products and network management tools.

Chapter 2,**Object Programming Basics**, provides basic explanations of how Invensys control objects work in I/A Series MicroNet controllers, including how information (data) is processed. This chapter also provides overviews of stencils and how control objects are represented in WP Tech, and explains the different general categories of control object types.

Chapter 3,**Understanding Programming Boundaries**, explains the device-specific boundaries present when engineering an I/A Series MicroNet controller application. Boundaries are most visible as resource tags, which include available I/O points, MicroNet sensor attributes, controller schedule functions, and LONMARK profile or network variable items. Controller memory resources and logical “Custom objects” are also explained.

Chapter 4,**Anatomy of a Control Object**, explains the common characteristics of any Invensys control object, which include configuration properties, input properties, and output properties. Common object behavior relating to mixing data classes, inverted inputs, and prioritized inputs is also discussed. Material in this chapter supplements the individual control object descriptions provided in Chapter 5.

Chapter 5,**Control Objects**, lists the 80-plus control objects by stencil groupings and also provides an alphabetical list. Each control object is individually described, with each description alphabetically sorted for easy reference. Each object description includes the WP Tech object representation (shape), device support and memory requirements, and tables describing the object’s configuration properties, inputs, and outputs. Each object description has an “Applying the Object” section to explain the object’s behavior and provides examples.

Chapter 6, **LONWORKS Network Data Exchange**, explains LONWORKS network variables (NVs) used in I/A Series MicroNet controllers. It begins with a general discussion of SNVTs. Then, it describes how SNVTs are implemented in the MicroNet standard controllers (profile tags) and the MicroNet MN 800 controller (SNVT objects). This is followed by individual descriptions of the NCI, NVI, and NVO objects. The last sections discuss the point history logging function, and how bindings are used to share data over a network.

Appendix A,**Memory Requirements**, explains the memory usage of control objects in I/A Series MicroNet controllers, including an alphabetical listing of all objects with their corresponding memory requirements.

Appendix B,**Profiles and Network Data**, provides quick reference information on each of the HVAC LONMARK Functional Profiles as implemented in I/A Series MicroNet standard controllers. This information includes details on Standard Network Variable Types (SNVTs) used in Invensys LONMARK profiles, and how SNVTs exchange data with MicroNet control logic. Next, reference information is provided on the MN 800 controller’s User-Definable Profile. The last section provides other

LONMARK/LONWORKS information applicable to MicroNet controllers, including Service Pin switch functions, default device addresses, and LONWORKS message services. These details are useful when using a LONWORKS network management tool.

Chapter 1
Introduction

This chapter provides a brief overview of the various I/A Series hardware and software products closely related to the WorkPlace Tech Tool, including:

- MicroNet Standard Controllers (MN 50, 100, 110, 130, 150, 200, and VAV)
- MicroNet MN 800 Controller (MN 800)
- MicroNet Digital Wall Sensors (MN-Sx Series)
- WorkPlace Communications Adapters
- The WorkPlace Tech Tool
- MicroNet VAV Flow Balance Software (see *MicroNet VAV Flow Balance User's Manual*, F-26421.

Finally, mention is made of various "third-party" LONWORKS based products. Both hardware products and software products exist in this category. The most notable software products are "network management tools", meaning PC programs used to logically configure a network of nodes (including I/A Series MicroNet controllers).

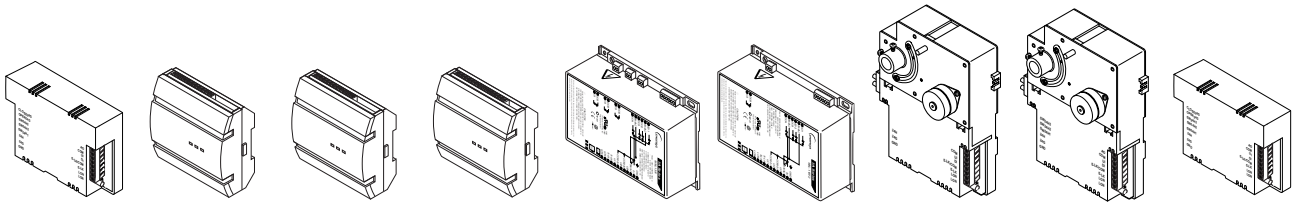
I/A Series Hardware Products

I/A Series hardware products include controllers, digital sensors, and WorkPlace communication (PC to LONTALK) adapter cards.

- I/A Series MicroNet standard controllers provide direct-digital control for fan coil, heat pump, roof top unit, unit ventilator, and VAV applications. Nine basic controller platforms are presently available; each has a number of I/O points and support for a digital room temperature or temperature and humidity sensor (I/A Series MicroNet sensor). Each controller contains a LONMARK HVAC Functional Profile that defines its "network image".
- I/A Series MN 800 controllers feature more I/O points than I/A Series MicroNet standard controllers, and use a programmable LONWORKS profile. The MN 800 has an integral real time clock (RTC) and also supports an I/A Series MicroNet sensor.
- I/A Series MicroNet sensors are digital sensors specifically for use with I/A Series MicroNet controllers. Twelve different models offer temperature only or temperature and humidity sensing with varying levels of sensor push-buttons and LCD screens.
- WorkPlace Communication Adapters are Echelon LONTALK PC adapter cards with integral FTT-10 transceivers. A PC running WP Tech requires an adapter to communicate with I/A Series MicroNet controllers.

MicroNet Standard Controllers

There are nine hardware platforms for I/A Series MicroNet standard controllers. Six platforms (MN 50, 100, 110, 130, 150, and 200) are for control of packaged rooftops, heat pumps, fan coils, and similar unitary applications. Three platforms (V1R, V2R, and V3R) are variable air volume (VAV) models.



Figure–1.1 I/A Series MicroNet Standard Controller Hardware Platforms.

Common Controller Features

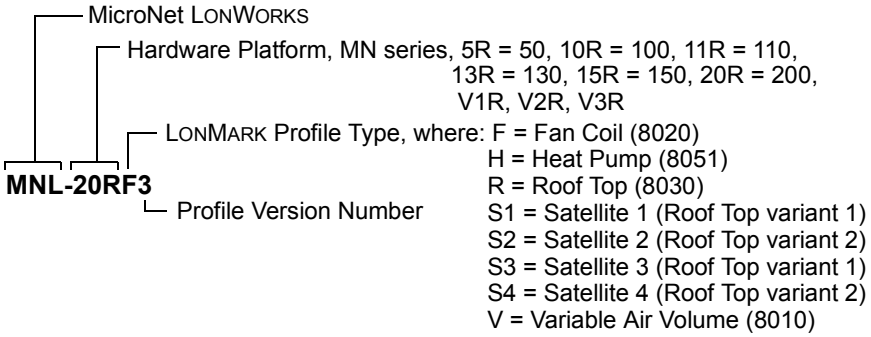
All controller platforms differ by physical characteristics and numbers and types of I/O points, but each controller platform provides these features:

- 24Vac powered.
- Standalone control capability.
- Support for a digital MicroNet sensor via a Sensor Link (S-LK) bus.
- LONMARK compliance, each having a LONMARK HVAC Profile.
- Onboard LONWORKS FTT-10 transceiver.

- Onboard LON Service LED, Receive and Transmit Data LEDs, and Service Pin Button.

Model Number Code

The model number for any MicroNet standard controller is coded with the following information:


MicroNet LONWORKS
Hardware Platform, MN series, 5R = 50, 10R = 100, 11R = 110,
13R = 130, 15R = 150, 20R = 200,
V1R, V2R, V3R
LONMARK Profile Type, where: F = Fan Coil (8020)
H = Heat Pump (8051)
R = Roof Top (8030)
S1 = Satellite 1 (Roof Top variant 1)
S2 = Satellite 2 (Roof Top variant 2)
S3 = Satellite 3 (Roof Top variant 1)
S4 = Satellite 4 (Roof Top variant 2)
V = Variable Air Volume (8010)
MNL-20RF3
Profile Version Number

Refer also to the section "Controller Firmware Revisions" on page 7.

Universal Inputs

The universal input characteristics are software-configured to respond to one of the five input types listed in Table–1.1.

Table–1.1 Input Types for Software-Configured Universal Inputs.

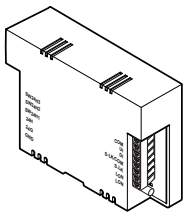
Input Type	Description
10K Thermistor with 11K Shunt Resistor	Sensor operating range -40 to 250 °F (-40 to 121 °C)[a]
Resistive	1 kOhm (130 to 950 ohm setpoint adjuster)
Analog Voltage	Range 0 to 5 Vdc
Analog Current	Range 4 to 20 mA[b]
Digital	Dry switched contact (for occupancy state, proof of flow, low limit, smoke, etc.)

a.Invensys model TSMN-57011-850 series or equivalent (for sensing space temperature).
b.An external 250 Ω shunt resistor is required.

MN 50, 100,150, 200 Hardware Platforms

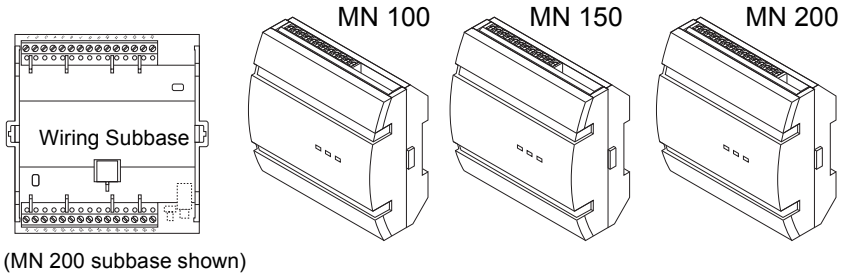
The MN 50 series is the smallest platform and has the fewest I/O points. An MN 50 controller can be panel mounted and has connections for power, communications, and I/O wiring on two terminal blocks. Refer to Table–1.2 below for the I/O points provided by the MN 50.

MN 50



Figure–1.2 MN 50 Controller Hardware Platform.

The MN 100, 150, and 200 hardware platforms feature a detachable subbase with screw terminals for wiring, and can be DIN rail or panel mounted. The electronics/cover assembly plugs into the wiring subbase. The controller cover has three status LEDs Figure-6.19, plus two hinged-flaps for accessing field wiring. Controllers provide a built-in LON Jack to allow local PC access to the LON. All three controller platforms have the same physical size and appearance, however, the MN 100 platform has fewest I/O points and the MN 200 platform the most I/O points. Refer to Table–1.2 below.



Figure–1.3 MN 100, 150, 200 Controller Hardware Platforms.

Table–1.2 I/O Point Comparison of MN 50, 100, 150, 200 Controller Platforms.

Type of I/O Point	MN **50** (MNL-5Rx)	MN **100** (MNL-10Rx)	MN **150** (MNL-15Rx)	MN **200** (MNL-20Rx)
Digital Inputs (DI)	1	1	—	2
Universal Inputs (UI)	1	2	3	3
Digital Outputs (DO)	3	4	2	6
Analog Output (AO)	—	—	2	2

LONMARK Profiles

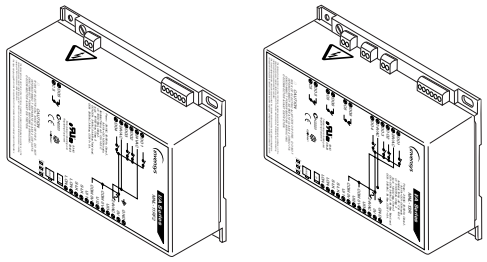
Each of the four controller platforms is available in models that differ by LONMARK profile, which is programmed at the time of manufacture. The profile is in read-only-memory and cannot be erased or changed. The following LONMARK profile types (and corresponding model numbers) for MN 50, 100, 150, and 200 are available:

Table–1.3 Available LONMARK Profiles and Model Numbers

LONMARK Profile	MN 50 (MNL-5Rxx)	MN 100 (MNL-10Rxx)	MN 150 (MNL-20Rxx)	MN 200 (MNL-20Rxx)
Fan Coil	MNL-5RF3	MNL-10RF3	MNL-15RF3	MNL-20RF3
Heat Pump	MNL-5RH3	MNL-10RH3	MNL-15RH3	MNL-20RH3
Roof Top Unit	MNL-5RR3	MNL-10RR3	MNL-15RR3	MNL-20RR3
Satellite 3	MNL-5RS3	MNL-10RS3	MNL-15RS3	MNL-20RS3
Satellite 4	MNL-5RS4	MNL-10RS4	MNL-15RS4	MNL-20RS4

Note: “Satellite” profile controller models are similar to those with Roof Top Unit profiles, but include a more flexible assortment of network variables for use in general-purpose types of applications. Refer to “Invensys LONMARK Profiles (page 616)” in Appendix B for more details on the LONMARK profiles above.

MN 110, 130 Hardware Platform



MN 110 Controller MN 130 Controller

Figure–1.4 MN 110, 130 Controller Hardware Platform.

The MN 110 and MN 130 Fan Coil Controllers with High Voltage Relays are provided with the LONMARK Fan Coil Unit functional profile (8020). They are programmed using the WP Tech to provide control for fan coil applications.

Table–1.4 Available LONMARK Profiles and Model Numbers

LONMARK Profile	MN 110 (MNL-11RFx)	MN 130 (MNL-13RFx)
Fan Coil	MNL-11RF3	MNL-13RF3

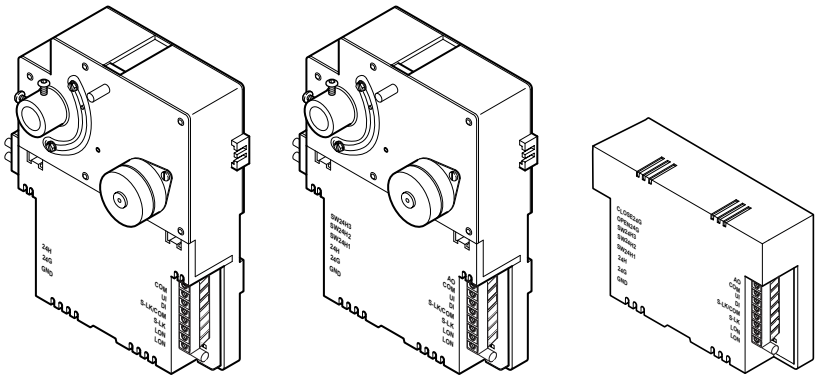
The MN 110 and 130 can be either DIN rail or panel mounted, and have terminal blocks for connections to power, communications, and I/O wiring. The MN 110 and 130 feature 3 A maximum 240 Vac relays making them especially suited for the fan coil application. The two models differ by the number of relays. Refer to Table–1.5. In addition, a built-in LON jack allows local PC access to the LON. Status indication is provided by three LEDs that can be viewed with the cover in place Figure-6.19.

Table–1.5 I/O Points on MN 110 and MN 130 Controllers

I/O Point Type	MN 110 (MNL-11RFx)	MN 130 (MNL-13RFx)
Universal Inputs (UI)	3	3
High Voltage Relay Outputs (DO)	1	3
24 Vac Triac Outputs (DO)	4	4

VAV Hardware Platforms

There are three different models of I/A Series MicroNet VAV controllers. All models have the LONMARK VAV Controller functional profile. Each model also has a built-in differential pressure transducer for measurement of velocity pressure (airflow) at a VAV terminal box, plus some additional points of I/O on field wiring terminal blocks. Status indication is provided by three LEDs Figure-6.19.



MNL-V1Rx MNL-V2Rx MNL-V3Rx

Table–1.6 Available LONMARK Profiles and Model Numbers

LONMARK Profile	V1R (MNL-V1RVx)	V2R (MNL-V2RVx)	V3R (MNL-V3RVx)
VAV	MNL-V1RV3	MNL-V2RV3‘	MNL-V3RV3

Figure–1.5 I/A Series MicroNet VAV Controller Hardware Platforms.

Models With Integral Actuators

Two VAV controller models have an integral actuator for over-the-shaft mounting on a damper of a VAV terminal box. The actuator tightens on a damper shaft using set screws, and a manual override button allows physical repositioning. Both of these controller models use the same physical package, but vary in numbers of I/O points Table–1.7.

Table–1.7 I/O Points for MNL-V1RVx and MNL-V2RVx VAV Controller Models.

Type of I/O Point	MNL-V1RVx	MNL-V2RVx
Digital Inputs (DI)	1	1
Universal Inputs (UI)	1	1
Digital Outputs (DO)	—	3
Analog Output (AO)	—	1

Model Without An Integral Actuator

The MNL-V3RVx series VAV controller has no integral damper actuator and so is physically smaller. This controller has two triac outputs for use with an external actuator, plus additional I/O points. The total number of I/O points are shown below Table–1.8.

Table–1.8 I/O Points for an MNL-V3R VAV Controller Model.

Type of I/O Point	MNL-V3R
Digital Inputs (DI)	1
Universal Inputs (UI)	1
Digital Outputs (DO)	3
Analog Output (AO)	1
Triac Outputs (TO)	2

Controller Firmware Revisions

The MicroNet standard controllers come with Revision 4.X (Rev.4.X) firmware. The MN 800 controller comes with Revision 1.X (Rev.1.X) firmware (or higher). Distinctions between controllers by firmware are noted throughout this manual because of the variation in functions provided with the different firmware revisions.

Identifying pre-Rev.3 Models

These controllers have pre-Rev.3 firmware:

- MNL-10Rx1 and MNL-20Rx1, where “x” denotes profile type (F, H, R)
- MNL-V1RV1, MNL-V2RV1, MNL-V3RV1

Identifying Rev.3.X Models

The following controller models contain Rev.3 or higher firmware:

- Where xx is F2, R2, H2, S1, or S2:
  - MNL-5Rxx
  - MNL-10Rxx
  - MNL-15Rxx
  - MNL-20Rxx
  - MNL-11RF2
  - MNL-13RF2
  - MNL-V1RV2
  - MNL-V2RV2
  - MNL-V3RV2

Identifying Rev.4.X Models

The following controller models contain Rev.4.1 or higher firmware:

- Where xx is F3, R3, H3, S3 or S4:
  - MNL-5Rxx
  - MNL-10Rxx
  - MNL-15Rxx
  - MNL-20Rxx
  - MNL-11RF3
  - MNL-13RF3

– MNL-V1RV3
– MNL-V2RV3
– MNL-V3RV3

Table–1.9 WPT Version Use With Controller Firmware

WorkPlace Tech Version	Standard Controllers with Rev 2.0 or 3.x Firmware	MN 800 Controllers	MNL-110 Controllers	MNL-130 Controllers	Standard Controllers with Rev. 4.1 Firmware
WPT 3.0	Yes	No	No	No	No
WPT 3.1	Yes	Yes	No	No	No
WPT 3.2	Yes	Yes	Yes	No	No
WPT 3.2 Service Pack 1	Yes	Yes	Yes	Yes	No
WPT 4.0	Yes	Yes	Yes	Yes	Yes

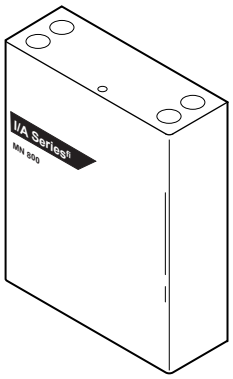
Profile Version Numbers

Table–1.10 Invensys LONMARK profiles Revisions

LonMark profile revision	Controller firmware revision	Profile changes
1	Pre-Rev. 3	Original release including Fan Coil F1, Heat Pump H1, Rooftop R1, and VAV V1.
2	Rev. 3	Version 2 profile release including Fan Coil F2, Heat Pump H2, Rooftop R2, and VAV V2. • SEC_tod_event was changed to a SNVT, SNVT_tod_event. • SEC_alarm was changed to a SNVT, SNVT_str_asc. • Added Satellite profiles S1 and S2 to the Invensys profile family.
3	Rev. 4	Version 3 profile release including Fan Coil F3, Heat Pump H3, Rooftop R3, and VAV V3 • nciSEC_model_num was changed to nvoDeviceInfo. • nvoSECAlarm was changed to nvoDeviceAlarm. • Added Satellite profiles S3 and S4 to the Invensys profile family.

Additional information can be found in Appendix B, “Profiles and Network Data (page 615)”.

MicroNet MN 800 Controller



Figure–1.6 MN 800 Controller Hardware Platform.

The MicroNet MN 800 controller differs from the MicroNet standard controller models by having a “programmable” LONWORKS profile, versus a fixed LONMARK profile, and by offering more I/O points. This controller features an onboard, capacitor-backed Real Time Clock (RTC) and provides additional capacities and features for more control functions.

Table–1.11 Model Chart for MN 800 Controller.

Model	Description
MNL-800-101	Controller Card with Backed-up Time Clock
ENCL-MZ800-WAL	Wall-Mount Enclosure
ENCL-MZ800-PAN	Panel-Mount Enclosure

Table–1.12 I/O Points for MN 800 Controller.

Type of I/O Point	Number of Points
Digital Inputs (DI)	—
Universal Inputs (UI)	8
Digital Outputs (DO)	8
Analog Output (AO)	4

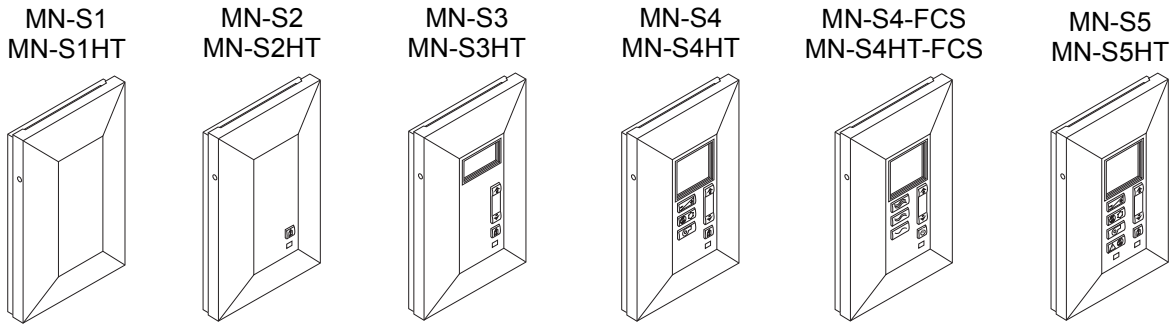
The MN 800 features the following:

- LONWORKS-compatible applications are completely programmable.
- A backed-up time clock provides true stand-alone direct digital control with optimum start stop, scheduling functions, and backed-up RAM.
- Programmable point history log (auto trending) with adjustable sample rates, continually accumulating log data, and a time stamp of the last 48 analog values or digital changes of state.
- One Universal Input (UI1) may be used for high speed pulse counting. Maximum pulse count rate is 10 per second with 50% duty cycle. All other UIs can be configured as DIs or for pulse counting with a maximum pulse rate of 1 per second with a 50% duty cycle.
- Functions as part of a LONWORKS FTT-10 Free Topology network.
- Support for one digital MicroNet sensor via the Sensor Link (S-LK) bus.
- LED indication of communications, service pin, LON jack.
- Controller card can be installed in either a wall-mount enclosure or a panel-mount enclosure. The controller card can also be plugged directly into an existing MicroZone II controller sub-base for upgrade to a LON.

- Eight universal inputs and eight pilot duty digital outputs

MicroNet Digital Wall Sensors

Any I/A Series LONMARK MicroNet controller supports a single digital wall temperature sensor. Twelve models are available with features ranging from temperature sensing only with no control or display to temperature and humidity sensing with a seven button control panel and LCD display.



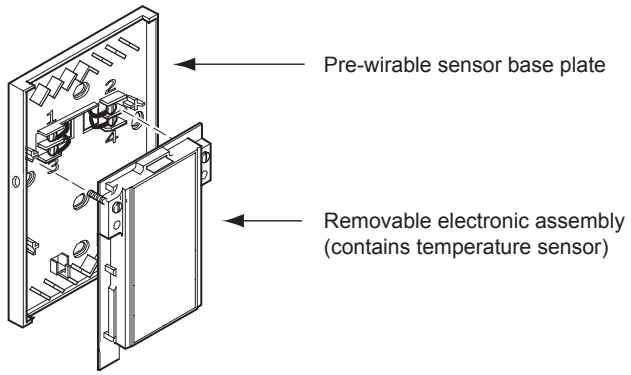
Figure–1.7 I/A Series MicroNet Sensor Models are MN-S1xx through MN-S5xx.

Common Sensor Features

An MN-Sxx sensor communicates with (and is powered) by two S-Link (S-LK) terminals on a MicroNet controller — it does not consume a typical I/O point. This connection between the sensor and controller can use low-cost twisted-pair wire up to 200 ft. (61 m), and is not polarity sensitive.

All MN-Sxx sensor models include a LON Jack to support an optional (and additional) twisted-pair connection to the LON. The LON Jack is compatible with the plug-in cable included with any of the three Invensys WorkPlace Communication Adapters (Echelon LONTALK PC adapters); Invensys models WPA-LON-1, WPA-LON-2, and WPA-LON-3.

Under the sensor's detachable cover, each MN sensor model includes a pre-wirable baseplate and a removable electronic assembly Figure-1.8. The same baseplate is used for each MN sensor model.



Figure–1.8 MN Sensor Pre-Wirable Baseplate and Electronic Assembly.

Note: MN Sensors have no independent intelligence. This means any MN sensor's behavior is defined by how the application control logic has been engineered, compiled, and downloaded into the MicroNet controller. This allows replacement of a sensor without need of additional programming.

Sensor Models and Functions

Table–1.13 shows the features available with each model of I/A Series MicroNet sensor.

Table–1.13 I/A Series MicroNet Sensor Features

I/A Series MicroNet Sensor Model	Description	Features								
		Zone Temp Sensing	Zone Humidity Sensing	Override Key and LED	Setpoint Adjustment	Fan Operation and Speed	Mode (Heat/Cool Auto/Off)	Emergency Heat Key and LED	LONWORKS Network Jack	Display Screen
MN-S1	MN-S1 has no display or keypad. Its primary function is to provide zone temperature to the controller via the S-Link. Provides a LONWORKS Network Jack for commissioning, testing, and monitoring.	X							X	
MN-S1HT	MN-S1HT adds humidity sensing functionality to the MN-S1.	X	X						X	
MN-S2	The MN-S2 provides zone temperature to the controller via the S-Link and features an Override Key, with LED indicator, which forces the controller into timed occupied mode. Provides a LONWORKS Network Jack for commissioning, testing, and monitoring.	X		X					X	
MN-S2HT	MN-S2HT adds humidity sensing functionality to the MN-S2.	X	X	X					X	
MN-S3	The MN-S3 provides the same functionality and features as the MN-S2. In addition, the MN-S3 has a digital liquid crystal display and allows controller setpoint adjustment. The MN-S3 offers one setpoint and one default display screen.	X		X	X				X	X
MN-S3HT	MN-S3HT adds humidity sensing functionality to the MN-S3.	X	X	X	X				X	X
MN-S4	The MN-S4 provides the same functionality and features as the MN-S3. In addition, the MN-S4 includes a Fan Key, a Mode Key, and a Setpoint Key. The keypad allows you to select controller modes, fan modes, and fan speeds. The MN-S4 offers four setpoints and four display screens.	X		X	X	X	X		X	X
MN-S4HT	MN-S4HT adds humidity sensing functionality to the MN-S4.	X	X	X	X	X	X		X	X
MN-S4-FCS	The MN-S4-FCS has a digital liquid crystal display and allows adjustment of one controller setpoint and display of one controller value. In addition, the keypad includes a Fan Key for On/Off/Auto settings and three Fan Speed keys for Low, Medium, High adjustment.	X			X	X			X	X

Table–1.13 I/A Series MicroNet Sensor Features (Continued)

I/A Series MicroNet Sensor Model	Description	Features								
		Zone Temp Sensing	Zone Humidity Sensing	Override Key and LED	Setpoint Adjustment	Fan Operation and Speed	Mode (Heat/Cool Auto/Off)	Emergency Heat Key and LED	LONWORKS Network Jack	Display Screen
MN-S4HT-FCS	MN-S4HT-FCS adds humidity sensing functionality to the MN-S4-FCS.	X	X		X	X			X	X
MN-S5	The MN-S5 provides the same functionality and features as the MN-S4. In addition, the MN-S5 features an Emergency Heat Key and LED for heat pump applications. The MN-S5 offers four setpoints and four display screens.	X		X	X	X	X	X	X	X
MN-S5HT	MN-S5HT adds humidity sensing functionality to the MN-S5.	X	X	X	X	X	X	X	X	X

Diagnostic Functions of LCD Screen Models

MN-S3xx, S4xx, S4xx-FCS, and S5xx sensors provide access to additional diagnostic data from a sensor-user keypad request. This Diagnostic Mode data is displayed on the LCD screens of these sensors, and includes separate displays (frames) for the MicroNet controller's:

- Subnet and Node Address
- Errors
- Alarms (see following Note)
- Temperature Offset
- Relative Humidity Offset

Note: Alarm frames allow viewing of the last four alarms in the MicroNet controller's local alarm buffer. These alarms correspond to Message ID numbers assigned in one or more control objects in the MicroNet controller. For more details, refer to the Chapter 5 examples included for the Analog Alarm object (page 121) and the Binary Alarm object (page 159).

With the exception of the Temperature and Relative Humidity Offsets, Diagnostic Mode data is view only. The offsets are adjustable and apply only to the integral temperature and relative humidity sensors in the MN sensor.

See the *I/A Series MicroNet Sensors General Instructions*, F-26277, for detailed information on the features and operation of MN sensors, including the Diagnostic Mode.

WorkPlace Communications Adapters

There are three models of WorkPlace Communications Adapters:

- WPA-LON-1 - An ISA adapter card for use in a desktop style PC.
- WPA-LON-2 - A Type II PC Card (formerly PCMCIA) for use in a notebook or laptop style PC.
- WPA-LON-3 - A PCI adapter card for use in a desktop style PC.

Common Adapter Features

These adapters are Echelon-manufactured LONTALK PC adapter cards that feature:

- An integral FTT-10 transceiver.
- Plug-and-play capability with Microsoft Windows 2000/XP. (No jumpers or switches.)
- A special 6 ft. (1.83 m) cable for plug-in connection to a built-in LON Jack featured on MN 100, 110, 130, 150, 200, and 800 controllers and all I/A Series MicroNet sensors.
- Compatibility with software drivers included with WP Tech.

WPA-LON-1

The WPA-LON-1 is Echelon's model PCLTA-10 PC LONTALK Adapter. This is a half-length, half-height card that requires an available 16-bit ISA slot in a PC for installation. The adapter has a removable two-position connector plug that can be terminated to the included 6 ft. LON Jack (Invensys) cable, or instead wired to any point on an FTT-10 based LON.

WPA-LON-2

The WPA-LON-2 is Echelon's model PCC-10 PC Card. This is a Type II PC Card with a special 15-pin Hirose connector for the network port. An included 6 ft. LON Jack (Invensys) cable connects this PCC-10 port to a built-in LON Jack featured on MN 100, 110, 130, 150, 200, and 800 controllers and all I/A Series MicroNet sensors.

WPA-LON-3

The WPA-LON-3 is Echelon's model PCLTA-20 PC LONTALK Adapter. This is a card designed for installation in an available 32-bit PCI slot in a PC. The adapter has a removable two-position connector plug that can be terminated to the included 6 ft. LON Jack (Invensys) cable, or instead wired to any point on an FTT-10 based LON.

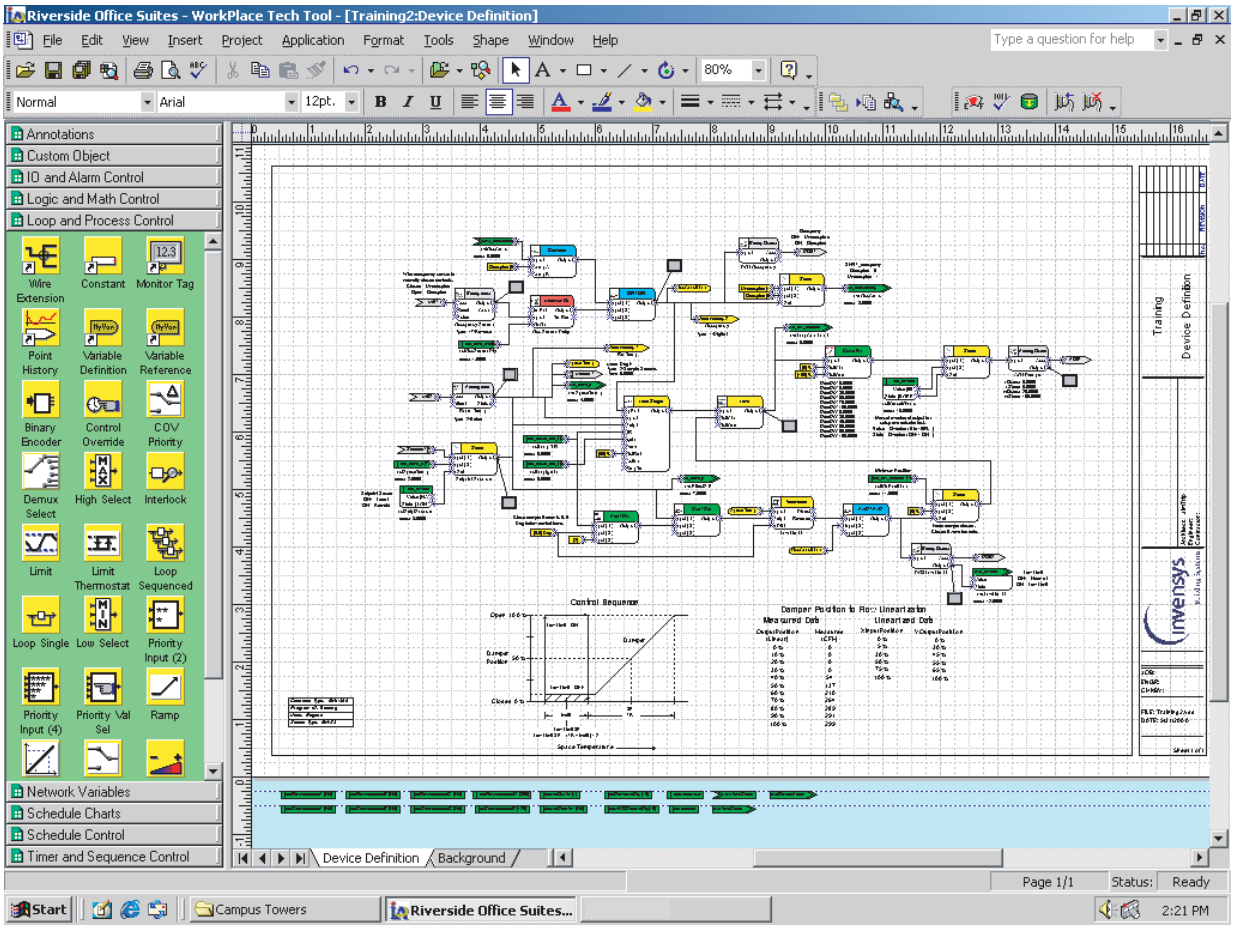
I/A Series Software Products

The WorkPlace Tech Tool

The WorkPlace Tech Tool (WP Tech) is the PC-based software tool used to program, compile, download, and upload and redraw a control application.WP Tech works with I/A Series MicroNet Controllers. WP Tech 4.0 is designed for use with Windows® 2000 Professional or Windows XP and Microsoft Visio® 2002.

Note: WP Tech 4.0 is not designed for use on with any other operating system, including Windows 98 or Windows NT®.

An application represents all the control logic in a controller. WP Tech uses a Visio™ 32-bit drawing interface to represent each application as a control logic drawing, using unique Invensys shapes for control objects and tags. Control objects are copied from stencils as needed, and have built-in “connection wires” that define the logic and flow of data in an application.



Figure–1.9 An Application is a Visio Drawing with a Control Logic Page.

Application Creation and Modification

Application creation and modification is done offline, with each application based on a special WP Tech (Visio) template. As each controller is fully programmable, fully custom control applications are possible. Alternately, a “pre-engineered” application can be used.

Pre-engineered Applications

A library of Invensys pre-engineered applications is available on the *I/A Series Application CD for WP Tech 4.0*. This CD may be obtained on request after contacting the Technical Services Group in the Invensys Customer Care Center. Note that these same pre-engineered applications are also available for download on the Invensys website, in a “zipped” format. Download access is password-protected.

Custom Application Overview

A “custom” Invensys application template provides a “blank” control logic page plus all of the Invensys control object stencils. The target MicroNet controller and sensor must be identified by running the Hardware Wizard. After the desired control objects are copied onto the drawing and connected as needed, the application can be compiled and downloaded into a MicroNet controller. In this way, a custom application can be built “from scratch.”

Application Upload

WorkPlace Tech 4.0 can upload application code from a controller and generate a Visio drawing from the uploaded information. This is very useful if an original drawing is lost or you must reverse engineer an application from an installed controller. Refer to the *WorkPlace Tech 4.0 User Guide*, Chapter 5 “Uploading Controller Applications” for detailed information about uploading applications.

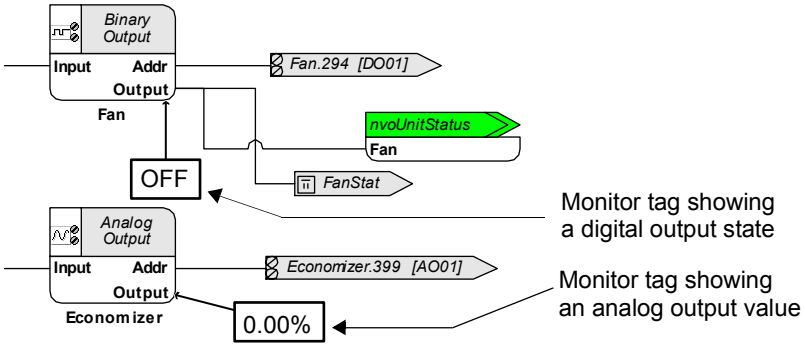
Applications that were created using WorkPlace Tech 4.0 include object positioning information from the original Visio drawings. WorkPlace Tech 4.0 uses this uploaded information to create drawings that are logically identical and very similar in appearance to the original drawings.

Applications that were created with WorkPlace Tech 3.2 or earlier do not include the original object positioning information. WorkPlace Tech 4.0 uses this uploaded information to create drawings that are logically identical to the original drawings. However, the appearance of the uploaded drawing will likely be very different than that of the original drawing.

Online Diagnostics

In addition to the download function, WP Tech provides an online “Connect” function that allows viewing of real-time data directly on an application’s control logic drawing. The application must first be downloaded into a selected I/A Series MicroNet controller.

Realtime data is received by moving “Monitor tag” shapes onto the drawing and attaching them to outputs of objects and tags. When “Connected” to an I/A Series MicroNet controller, values in the monitor tags update. Monitor tags are for de-bugging and not intended for extended real time monitoring.



Figure–1.10 Monitor Tags Allow Real-time Viewing of Controller Values.

Note: Connected (active) monitor tag values are polled, causing additional network traffic. Be sure to disconnect (de-activate) monitor tags to relieve network traffic burden.

Diagnostic Writes

Fixed values to control objects can be temporarily modified when connected to a controller, using a “Write to RAM” function. This allows quick changes to constant values when diagnosing an application online. Temporary values are cleared after resetting the controller or after downloading a database to the controller.

A separate but similar function is available to write or read values in configuration properties of the profile, which are called Network Configuration Inputs (NCIs). Any change to an NCI is permanently retained until it is changed again. Refer to “Input Profile Tags (NCIs and NVIs)” on page 543 and “NCI Objects (nciType)” on page 548.

Project Based Folders

In WP Tech, all work is done in a *project*. A project is a folder that contains one or more applications. Usually, a project is assigned the job’s name and the applications are assigned names to indicate which controllers they represent.

Each time WP Tech is started, a Projects Dialog Box lists all projects saved in the default Projects directory. For best organization, it is highly recommended that all projects be kept in the default Projects directory.

Note: It is easier to organize and maintain multiple small projects rather than a single large project.

MicroNet VAV Flow Balance Software

The MicroNet Flow Balance software is used to air balance terminal boxes controlled by I/A Series MicroNet VAV controllers.

The MicroNet VAV controllers are interoperable, LONMARK-compliant devices that provide a wide range of control strategies for pressure independent terminal boxes with, or without, reheat capabilities.

Third-Party LONWORKS Products

Manufacturers other than Invensys are also producing LONWORKS-based hardware and software. Typical hardware devices include special-purpose controllers, discrete I/O modules, sensors, and transducers. For the most part, these devices perform a fixed function and can be configured (but not programmed). Any required configuration is achieved by accessing configuration properties and network variables of the LONMARK objects or profile using a network management tool.

Third-party LONWORKS-based software programs are in two major categories:

- Network Management Tools
- User Interface Tools

Of the two, a network management tool is the most essential when installing a job with networked nodes, as it is needed for the logical configuration of any LONWORKS network (LON).

Network Management Tools

A network management tool is required to assign logical addresses to networked LONWORKS nodes (including I/A Series MicroNet controllers) and to make any subsequent “bindings” between network variables among the nodes. It can also be used to modify external configuration parameters of a node.

I/A Series WorkPlace Pro™ provides a comprehensive set of engineering tools combined into one common easy to use graphical-based engineering environment. WorkPlace Pro includes a network management tool for LONWORKS based devices including node installation, network variable bindings, and automatic network “learning.”

When MicroNet I/A Series controllers are used on an I/A Series Niagara Web-accessed system, WorkPlace Tech Tool can operate at a remote location by use of the Virtual LON (VLON) service. A computer operating with WorkPlace Tech Tool and VLON can provide full access to a MicroNet I/A controller at a different location via an ethernet connection and an I/A Series Universal Network Controller at the remote location.

Two network management tools are Echelon’s LONMAKER® for Windows and IEC’s ICELAN 2000™. Both programs present a graphical representation of a LON network. Functions are included for node installation, network variable binding, and network variable browsing.

Chapter 2
Object Programming Basics

This chapter provides basic explanations of how Invensys Control Objects work in the I/A Series MicroNet controllers, including how information (data) is processed. This chapter also provides an overview on how control objects are represented in WP Tech, and explains the different general categories of control object types.

The following topics are discussed in this chapter:

Objects in an I/A Series MicroNet Controller

- An Object as an Algorithm
- Object Inputs and Outputs
- Linked with Other Control Objects
- Configuration Properties
- Data (Number) System
    - Analog Data
    - Digital Data
    - Not Active (NA)
- Use of Controller Object Memory

Objects in WP Tech

- Shape Stencils
- Control Objects
- Tags
- Linking Objects (and Tags)
- Engineering Process Overview

Control Object Categories

- I/O Point Objects
- Functional Objects

Objects in an I/A Series MicroNet Controller

An I/A Series MicroNet controller is programmed using Invensys control objects. A typical controller operates with control objects of various *types*, selected as needed to perform a particular control *application*. An application includes all the control objects stored in (and processed by) a controller, and represents the control logic in a particular controller.

There are 80-plus different types of control objects available. Each object type performs a unique pre-defined function. Control object types can be grouped in one of two broad categories; I/O point objects and functional objects. A *few* types from each category are listed below (a complete list of all objects in each category is found near the end of this chapter):

- I/O point objects
  - Analog Input object
  - Analog Output object
  - Binary Input object
  - Binary Output object
- Functional objects
  - Loop Single and Loop Sequenced objects
  - Logic objects (for instance AND / OR, EXOR, OR / AND)
  - Math objects (Add / Div, Average, Mul / Div)
  - Timer objects (Dual Delay, Minimum On, Minimum Off)

Typically, an application uses a number of I/O point objects, that correspond to physical I/O points, and some number of functional (logic) objects. All of the control objects used in the application reside in the controller's non-volatile EEPROM memory, and operate in the controller's RAM memory.

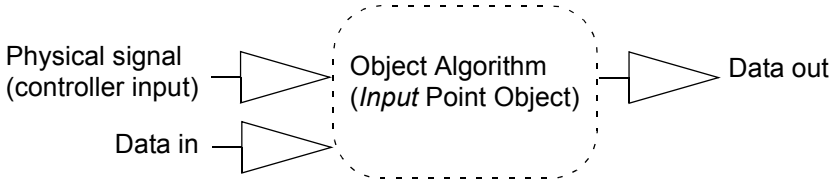
An Object as an Algorithm

A control object in an I/A Series MicroNet controller performs a pre-defined operation or *algorithm*, based on its object type and the data received and produced (as shown below). Data in and out is a numerical value that represents either a digital state, such as OFF or ON, or an analog level, such as temperature reading.

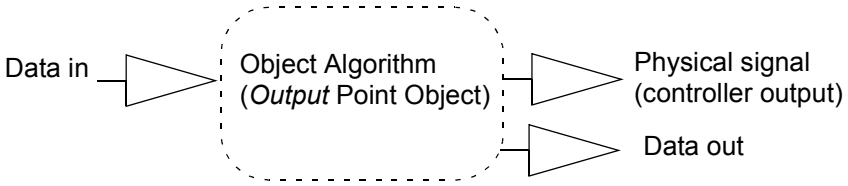


Figure–2.1 Object Algorithm Takes Data In and Produces Data Out.

If the control object is an I/O point type object, data also includes a physical (hardware) signal, such as a voltage read at a controller input or produced at a controller output, as illustrated in the two examples below.



Figure–2.2 Input Point Objects Read a Physical Signal at a Controller Input.



Figure–2.3 Output Point Objects Produce a Physical Signal at a Controller Output.

The algorithm is the actual *work* the control object does as executed by the controller's processor, whether producing a numerical value, or sampling (or generating) a physical signal. Algorithms of some object types are simple in nature; for example, the Add / Add object is a math type object that simply adds all values received and produces the sum result. Most object algorithms are more advanced and involve several different factors, including timed sequences.

Object Inputs and Outputs

Each control object has properties, which include one or more input properties (inputs) and one or more output properties (outputs).

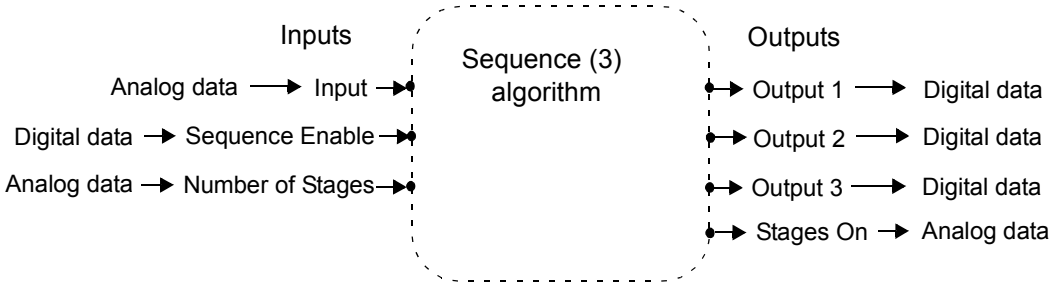
- An input allows the reception of data for use in the object's algorithm.
- An output provides the data results of the object's algorithm.

A control object's type determines the number of inputs and outputs. For instance, most logic and math type objects each have three inputs (for receiving data) and a single output (the data result).

Analog and Digital Inputs and Outputs

A control object's type also determines how individual inputs evaluate data (classified as analog or digital), and how each output formats the data results. For example, logic type objects evaluate all inputs as digital values and produce a digital output result; math type objects evaluate all inputs as analog values and produce an analog output result.

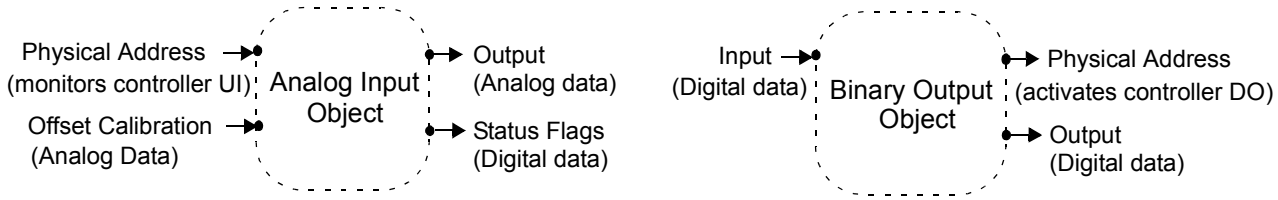
Many other control object types have a mix of digital inputs and analog inputs, and often a mix of digital outputs and analog outputs as well. For example, a Sequence (3) object has both analog and digital classes of inputs and outputs.



Figure–2.4 A Sequence (3) Object Has Both Analog and Digital Inputs and Outputs.

Physical Address Inputs and Outputs

I/O point type objects are used to interface to physical I/O points on a controller. These object types include special *Physical Address* inputs or outputs used to assign a particular I/O terminal address. For instance, an Analog Input object has a physical address *input* used to specify which of the controller's universal input (UI) terminals that it monitors. A Binary Output object has a physical address *output* that determines which of the controller's digital output (DO) terminals that it activates.



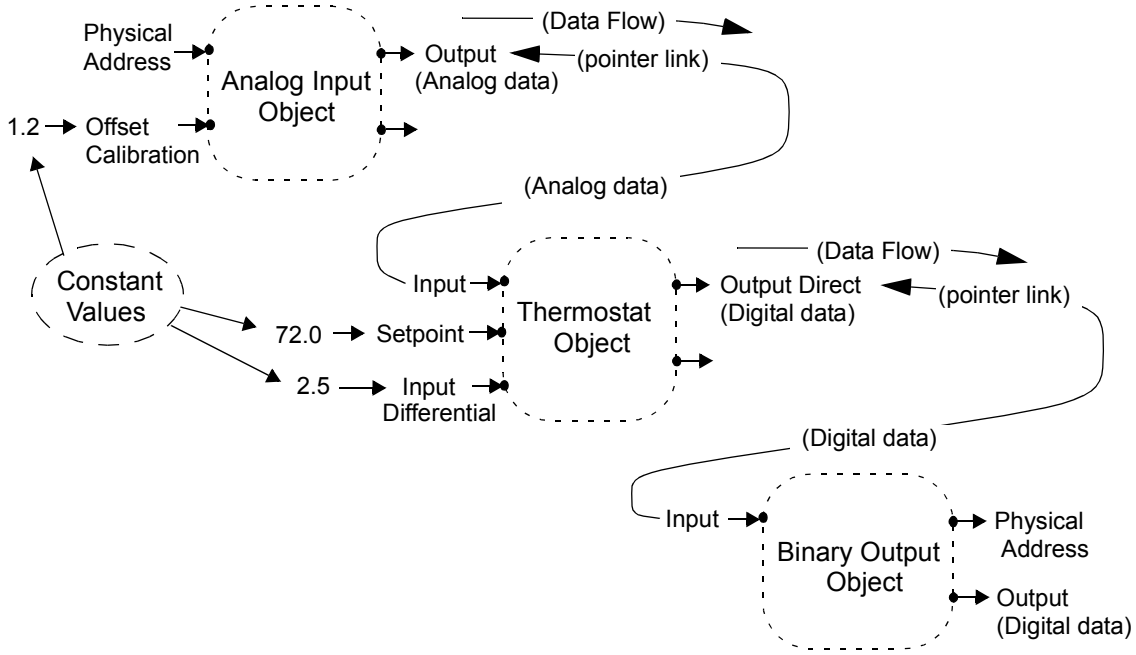
Figure–2.5 I/O Point Type Objects Include a Physical Address Input or Output.

Linked with Other Control Objects

Apart from physical address inputs, any input on a control object can be assigned a constant value or linked (via a pointer) back to an output of another control object. If linked to an output with a pointer, the input continuously tracks the output value. Linking control objects together is central to creating control logic, allowing well-defined control sequences.

As an example Figure-2.6, three control objects are linked together:

- The Analog Input object (top) outputs an analog value based on the physical signal monitored at the controller universal input (UI).
- The Thermostat object (middle) has its primary input point back to the output of the Analog Input object, and so receives the analog data. The Thermostat object's algorithm compares this input value to the value at its Setpoint input (in this case a constant value of 72.0), and outputs digital values as necessary at its outputs (including Output Direct).
- The Binary Output object (bottom) has its input point back to the output of the Thermostat object's output, and so receives its digital data. The Binary Output object activates the assigned controller digital output (DO) to ON whenever its input receives an ON from the Thermostat object.



Figure–2.6 Example Control Objects Linked with Pointers.

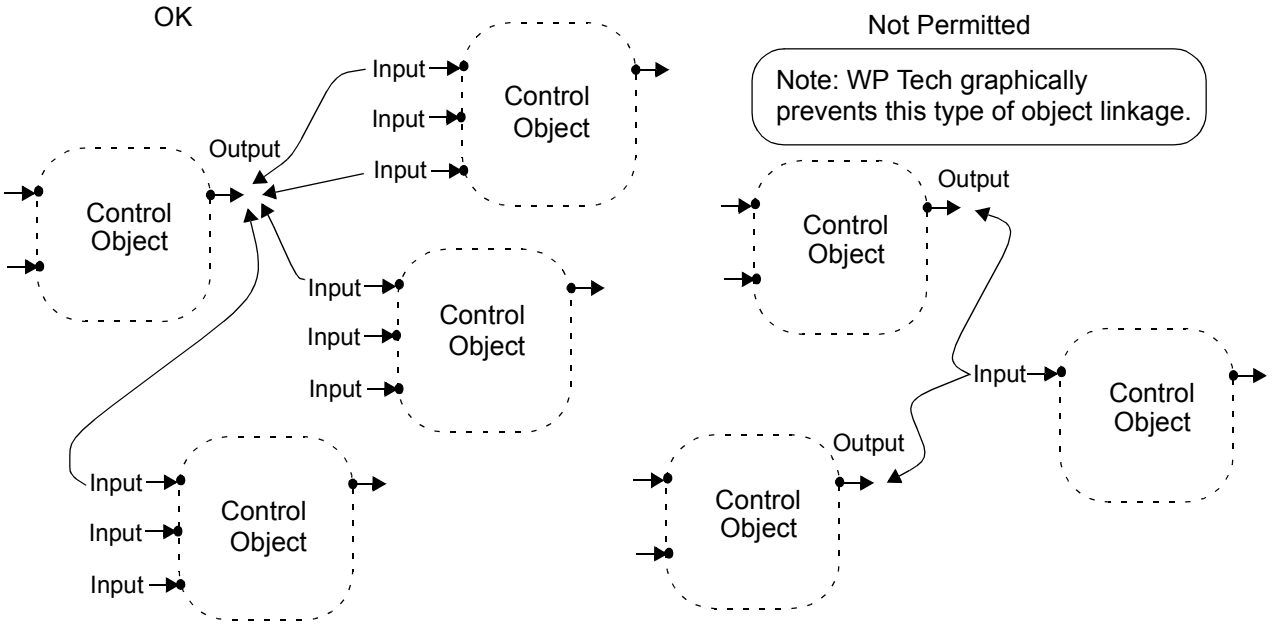
The two characteristics of objects linked by a pointer are:

- The pointer link is from input (back) to output.
- Data flow is from output to input.

In the example above, additional control objects and pointers could be easily added. For example, instead of a constant value of 72.0, the Setpoint input to the Thermostat object could point to an output of a Reset object, which in turn is linked to another Analog Input object.

One Output to Many Inputs

More than one input of a control object can be linked to one control object output. In fact, as many inputs of as many control objects as needed can be linked to a single object output (one output to many inputs). However, note that the reverse is not true—an input can only be sourced from one destination; either (one) output of another object or from a constant value. Compare the two types of linkages below Figure-2.7.



Figure–2.7 One-to-many is OK, many-to-one is NOT permitted.

Configuration Properties

In addition to input properties (inputs) and output properties (outputs), each control object also has *configuration* properties. Three common configuration properties are used in every object type, which serve to:

- Identify each particular control object in a controller with user-defined labels (Object Name and Description). Descriptions are not uploaded with an application.
- Determine the frequency of the object's execution in the controller (Process Time, standard controllers only).

Configuration properties in many object types are limited to just the common ones above. However, many other object types have additional configuration properties that affect how the object's particular algorithm operates. These can be considered *algorithm-related* configuration properties.

For instance, an Analog Input object has a Type configuration property that defines what sort of input sensing device (sensor) is wired to the controller's UI terminal that is monitored by the object. The Type property selection determines how the algorithm uses the physical signal at the UI, in order to help compute the correct value measured at the sensor.

Typically, a control object's configuration properties are not often changed after the object is created. Unlike inputs and outputs, configuration properties cannot be linked to other properties. Essentially these properties are constant values that are stored in the controller's non-volatile EEPROM.

Data (Number) System

All I/A Series MicroNet controllers process data numerically, whether the data represents an analog (level) or digital (status) type of information. Invalid or abnormal data is represented as a special not active (NA) value, which is evaluated differently depending on the object type.

Analog Data

The full range for any data value in a MicroNet controller is:

-163.83 (minimum) to 16,383 (maximum).

Analog values have a decimal component only in the “scaled” portion of this range, which is:

-163.83 to 163.83 (with a resolution of 0.01)

Any value above 163.83 operates as an integer, from 164 to 16,383.

By default, analog values operate in the scaled portion, that is, from -163.83 to 163.83. This format is used for temperatures, percents, pressures, and general math. The integer format is used when an analog value exceeds 163.83 or to represent enumerated values, counters, or units of time such as minutes.

Note:

- If an attempt is made to enter a value outside the full range of -163.83 to 16,383, WP Tech displays an *error message* indicating that the entry is not valid, and displays the valid range as a guide to the user.
- Be aware that *output values* may not always appear as expected, due to the way scaled and integer numbers are processed:
  - If the value is entered within *WP Tech*, using a *Constant tag*, WP Tech *truncates* the value before downloading it to the controller, while the value *displayed* in the Constant tag *remains unchanged*. A number in the scaled range (-163.83 to 163.83) is truncated to two decimal places. A number in the integer range is truncated to its whole number. Be sure to note this difference between the displayed value and the processed value when checking outputs. For example, a value of 189.66 that is entered in a Constant tag appears to be unchanged, while the actual value downloaded to or uploaded from the controller is truncated to 189.
  - If the value is *generated within the application*, such as an NV tag or a calculation within an object, the control logic automatically *rounds* the number. A number in the scaled range is rounded to within .01. A number in the integer range is rounded to the nearest whole number. If the value in the preceding example, 189.66, were a calculated value within the application, it would be rounded to 190.

Digital Data

Digital data has only two valid states, OFF or ON. Control objects with digital class outputs represent this numerically by producing an output value of either zero (0.0) for OFF, or 100.0 for ON.

Digital class inputs use a wider range to evaluate any received value as either OFF or ON as follows:

OFF is any value from -163.83 to 0.0 (less than or equal to zero).

ON is any value from 0.01 to 16,383 (any positive number).

Note: Representing a digital state numerically allows a pointer link from an analog type input to a digital output, and the opposite — a link from a digital type input to an analog type output.

Not Active (NA)

A not active value (NA) represents an undefined or null value. Any input to an object left unconnected (the default state, with no assigned constant value or pointer to an object output) is considered NA.

NA is handled differently by the algorithms of the various control objects. Depending on the particular type of control object, an object can produce an NA output if one or more inputs has an NA or if some abnormal condition occurs. With a few objects, an input with an NA is evaluated the same as zero (0.0) value. Other object types have inputs that ignore an NA input.

Refer to the control object descriptions for details on how NA values affect any particular control object, including under what circumstances the object produces a NA output (or outputs).

External Data Exchange in a MicroNet Controller

As a LONMARK or LONWORKS device, an I/A Series MicroNet controller can exchange data with other devices (nodes) on a LONWORKS network using network variables and SNVTs in the profile. Some SNVTs specify numerical ranges that exceed the internal data limits of the controller. Note, however, that all data processed by control objects in MicroNet controllers is limited to the full range of the numbering system, which is:

-163.83 (minimum) to 16,383 (maximum).

This means, for example, that if a data value enters the controller profile on a network variable input (NVI) as 31145, it will be evaluated by any control object input as 16383 — the highest possible value. For more details, refer to "Profiles and Network Data" section in Appendix B of this manual (page 615).

Use of Controller Object Memory

A controller cannot contain unlimited numbers of control objects—objects consume memory resources. However, there are no maximum limits or preset allocations as to specific types of objects. For example, a controller can have eight or more Loop Sequenced objects, as long as sufficient object memory remains for the other objects needed in the application.

Each control object created in an I/A Series MicroNet controller consumes a specific amount of non-volatile EEPROM and RAM type memory. The amount of object memory required depends on the object's type. For example:

Priority Input (2): 8 bytes of EEPROM, 10 bytes of RAM.

Interstage Delay (10): 68 bytes of EEPROM, 114 bytes of RAM.

The Interstage Delay (10) object consumes over eight times the EEPROM and eleven times the RAM as the Priority Input (2) object.
This difference is due to the complexity of the Interstage Delay (10) object's algorithm and its more numerous outputs. Refer to "Memory Requirements" section in Appendix A of this manual (page 609) for a complete listings of controller memory capacities and memory requirements for each control object type.

WP Tech includes an application “Statistics” function that displays assorted information about the active application, including the current memory resources available. The Statistics function is available under the Application pulldown menu. Using the Statistics function causes WorkPlace Tech to compile the application.

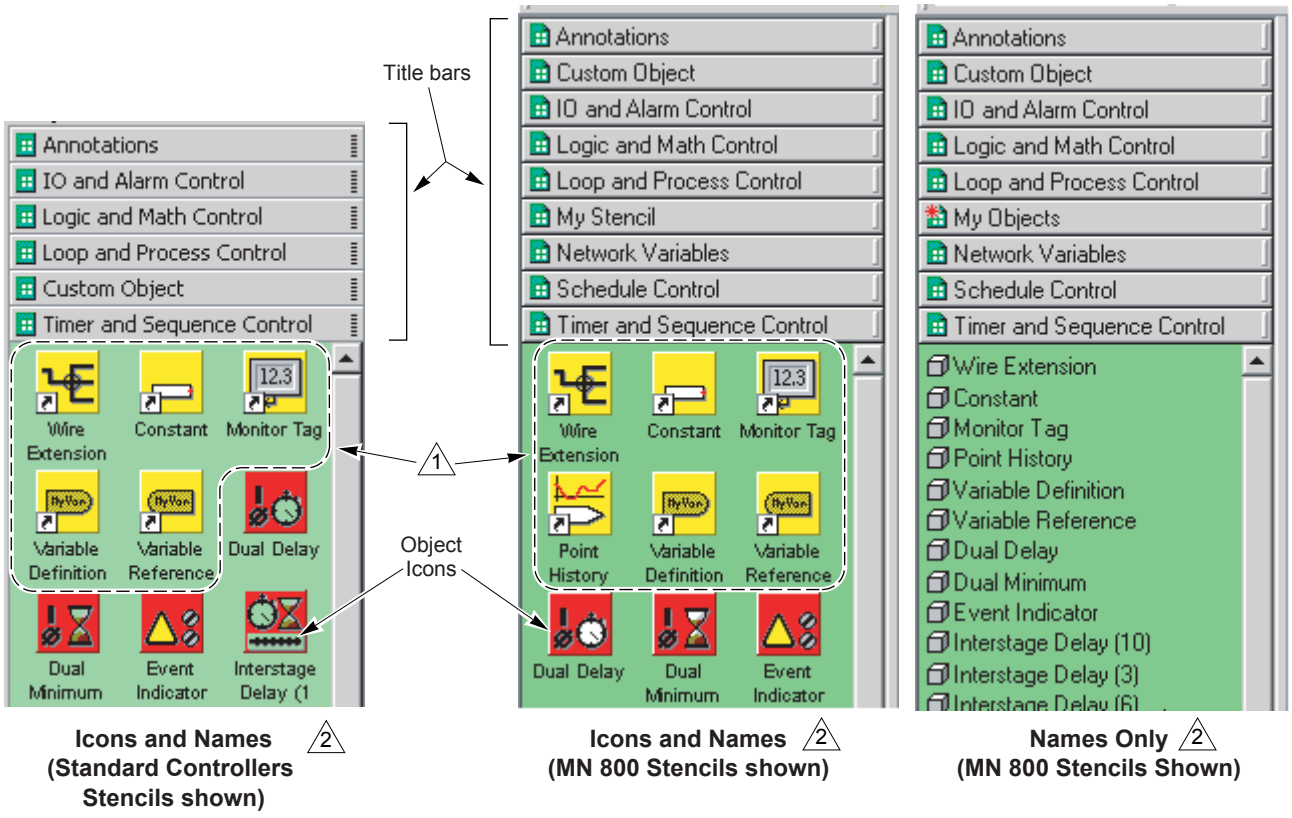
Objects in WP Tech

Control objects are programmed in I/A Series MicroNet controllers using the offline engineering tool WP Tech. This Visio-based program allows control logic to be graphically engineered on an application drawing. The application drawing can then be compiled into a hex file and downloaded into the target MicroNet controller, creating the equivalent control objects. WP Tech also provides online diagnostic functions that allow real-time monitoring of object’s outputs and temporary writes to constant values, again using the application drawing as a reference.

The WP Tech shape for each control object is included in the individual object descriptions in this reference. For detailed information on using the Visio-based WP Tech software program, refer to the *I/A Series WorkPlace Tech Tool User’s Guide*, F-26987.

Shape Stencils

As a Visio-based program, WP Tech contains a number of *stencils* Figure-2.8, which are collections of related Visio master shapes. With the exception of the Annotations stencil, the same shapes appear at the top of every stencil: Wire Extension, Constant, Monitor Tag, Point History (MN 800 only), Variable Definition, and Variable Reference. Any of the shapes on a stencil can be copied to a Visio drawing, using the drag-and-drop method.



1 Shapes appearing on all stencils (except Annotations).

2 There are four display options for stencils, selected by right-clicking on a stencil: Icons and Names (default), Icons Only (not shown), Names Only, and Icons and Details (not shown).

Figure–2.8 Stencils in WP Tech.

Stencils have four optional views: Icons and Names (default), Icons Only, Names Only, and Icons and Details (not shown). A different view may be selected by right-clicking on the stencil's title bar, then clicking the desired view. A click on a stencil's title bar opens the stencil to reveal the shapes contained within it. In each application, WP Tech displays those stencils and objects that are applicable to the specified hardware platform. Refer to the "Objects Grouped Alphabetically" section in Chapter 5 of this manual (page 105) for a complete list of the control objects and the hardware platforms on which they are supported.

Six of the WP Tech stencils are common to all applications. Two stencils, Network Variables and Schedule Control, are available only in MN 800 controller applications Figure-2.8.

WP Tech Stencils

The WP Tech stencils are described in the following subsections.

Annotations

This stencil contains shapes that may be used to easily add textual information to a drawing. Included are:

- A large selection of standard Visio annotation tools including callouts, text, balloons, stamps, tags, and starbursts.
- Separate WP Tech objects for placing on a drawing, an Invensys background, a controller information block, a sequence of operation, a link to a Microsoft Word document, or one of two variations of the Invensys logo.

Custom Object

This stencil contains three shapes used to create custom objects: a Custom Object, a Custom Input Tag, and a Custom Output Tag. The Custom Object shape on this stencil is used to create a custom object, which represents control logic that is defined by a group of interconnected shapes on an underlying page. Inputs and outputs are assigned to a custom object by applying two other shapes from this stencil, the Custom Input tag and the Custom Output tag. These tags are copied onto the definition page and connected to selected object inputs and outputs on that page. Any number of custom objects may be created and saved, on a user's stencil, for reuse in future applications.

IO and Alarm Control

These control objects may be used to detect alarm conditions and write unique alarm message IDs to the MicroNet controller's local alarm buffer. The Sensor Input object on this stencil is available only when creating MN 800 applications.

Logic and Math Control

The objects on this stencil contain logic or math functions, and are among the simpler of the functional objects. Most logic and math control objects have three inputs and a single output. The Enthalpy object on this stencil is available only when creating MN 800 applications.

Loop and Process Control

These control objects perform various functions relating to direct digital control (DDC), HVAC control, or priority handling of data. They range from simple objects, such as the Select object, to ones with complex control algorithms, such as the Loop Sequence object. The Ramp object on this stencil is available only when creating MN 800 applications.

Network Variables (MN 800 only)

This stencil contains three SNVT objects, NCI, NVI, and NVO, which are used to complement the MN 800 controller's mandatory SNVT objects, thereby customizing the controller's network image or profile. These objects offer extended functionality and flexibility over equivalent profile items in MicroNet standard controllers.

Schedule Control (MN 800 only)

Three control objects are featured on this stencil, the Calendar object, the OSS object, and the Schedule 7-Day object. The Calendar and OSS objects are used to schedule annual exception periods which require a change from normal operation, and to ensure that these changes occur with minimum energy usage. The Schedule 7-Day object provides a means for scheduling a seven day, repeating set of events.

In addition, this stencil features six clock tags: Year, Month, Day, Hour, Minute, and Second. These are input tags that provide current clock information to an application. For more information on these tags, refer to Clock Tags (MN 800).

Timer and Sequence Control

These control objects perform various time-based functions, including delays, minimum on or off periods, and output sequencing. The Step Driver object on this stencil is available only when creating MN 800 applications.

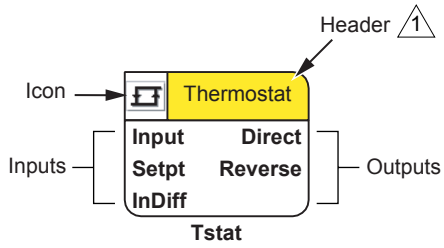
Creating New (Custom) Stencils

In addition to the stencils provided with WP Tech, you may create one or more custom stencils to store frequently used objects for easy access. These may include Custom objects, other WP Tech objects, and Visio objects. A My Solutions directory is provided in WP Tech, as a convenient place to store these custom stencils. Refer to the *I/A Series WorkPlace Tech Tool User's Guide*, F-27275, for step-by-step instructions on creating new stencils.

Control Objects

Each WP Tech control object stencil contains master copies of Invensys control objects that are graphically depicted as specialized Visio *shapes*. When an instance of a master shape is copied to the application drawing, using the drag-and-drop method, it appears as a box-shaped object having similar characteristics Figure-2.9:

- The *type* of control object is labeled at the top of the shape in a colored header Figure–2.1 and an icon representing the object type.
- *Inputs* are always shown on the *left* side of the shape. Abbreviations are often used, for example, InDiff for Input Differential.
- *Outputs* are always shown on the *right* side of the shape. Abbreviations are often used, for example, Direct for Output Direct.



Figure–2.9 A Control Object as Represented in WP Tech (Visio).

Table–2.1 Color Code for Shapes

Shape Type	Color
IO and Alarm Control Objects Resource Tags	Gray
Logic Objects	Light Blue
Math Control Objects Network Variable Objects and Tags	Green
Loop and Process Control Objects Support Tags	Yellow
Schedule Control Objects Timer and Sequence Control Objects	Red
Custom Objects	Black

Control objects are dragged and dropped from stencils onto the application drawing page, moved about, and configured on an as-needed basis, until enough objects exist to perform the required control sequence. Control objects can also be duplicated, deleted, or even copied from another WP Tech 4.0 application drawing.

Configuring a Control Object

Once a control object is created, it is configured by modifying its default properties settings or by changing its appearance. The input, output, and configuration properties of any selected object may be modified through the Properties Editor. The object's appearance may be modified by changing its name, or by showing or hiding its name, inputs, outputs, or properties, through the Customize Object dialog box. By default the object's name appears below it, and all its inputs and outputs are shown (with the single exception of the OSS object, as described in Additional Output Properties, on page 374).

Tags

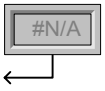
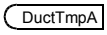
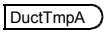
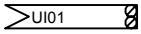
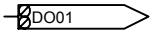
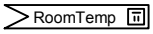
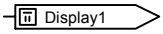
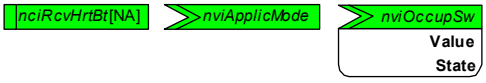
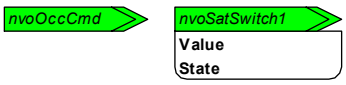
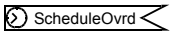
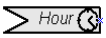
Apart from the 80-plus types of control objects, other Invensys shapes are used in WP Tech. These other shapes are called *Tags*, of which there are two main categories:

- Object Tags - like control objects, these shapes are copied from master shapes on control object stencils. Two types of input object tags are used to feed control object inputs; either a fixed value (Constant Tag) or a named variable (Variable Reference Tag). A corresponding output tag (Variable Definition Tag) allows an object output to be assigned to a variable name. A special case output tag called the Monitor Tag allows real-time checkout of an object's output (but has no direct effect on the control logic). Like control objects, object tags are used as needed in an application.
- Resource Tags - these shapes are generated from the Hardware Wizard of WP Tech and reside to the left, right, or bottom of the drawing. Collectively, these shapes represent all the physical resources of a selected MicroNet controller including controller I/O terminals, network profile components, attributes of a selected MicroNet sensor, plus the controller's software clock and schedule (software clock and schedule tags are not available in Rev. 2 controllers). Unlike control objects, resource tags *cannot* be duplicated or deleted when engineering control logic (but may be left unused).

Each tag has either a single output or a single input, depending on whether it is an input tag or an output tag.

- An input tag (such as a Constant Tag) has an output.
- An output tag (such as an Output Hardware Tag) has an input.
- An exception is the MicroNet sensor fan tag, which has two outputs.

Control objects use object tags and resource tags to help define the control logic used in the application. Some example input and output object tags and resource tags are shown in Figure–2.10.

<table>
<tr><th colspan="2">Object Tags from Stencils</th></tr>
<tr><th>Input Tags</th><th>Output Tags</th></tr>
<tr><td>Constant Tag:
  </td><td>Monitor Tag:
  </td></tr>
<tr><td>Variable Reference Tag:
  </td><td>Variable Definition Tag:
  </td></tr>
<tr><td></td><td>Point History (MN 800)
  </td></tr>
<tr><th colspan="2">Resource Tags from Hardware Wizard</th></tr>
<tr><th>Input Tags</th><th>Output Tags</th></tr>
<tr><td>Hardware Tag:
  </td><td>Hardware Tag:
  </td></tr>
<tr><td>Sensor Tag:
  </td><td>Sensor Tag:
  </td></tr>
<tr><td>Profile Tags:
  </td><td>Profile Tag:
 
Note: In the MN 800, NV objects are used in place of profile tags.</td></tr>
<tr><td>Schedule Tag
  </td><td>Schedule Tag
  </td></tr>
<tr><td>Clock Tag
  </td><td></td></tr>
</table>

Figure–2.10 Example Object Tags and Resource Tags in WP Tech.

Note:

- Refer to the *I/A Series WorkPlace Tech Tool User's Guide*, F-27255, for detailed information on using object tags and creating resource tags with the hardware wizard.
- In the MN 800, NV objects are used in place of profile tags.
- Sensor Tags, which are a special type of resource tag, determine the behavior of the S-Link Sensor and are similar to control objects.
- Schedule and Clock Tags, a special type of resource tag available in Rev.3 and higher standard controllers, represent the controller's built-in schedule and software clock.
- Sensor tags and schedule tags are explained in detail in Chapter 3, "Understanding Programming Boundaries", in sections "S-Link Sensor (Sensor Tags) (page 58)" and "Schedule Tags (page 75)".

Linking Objects (and Tags)

Rules for Connection

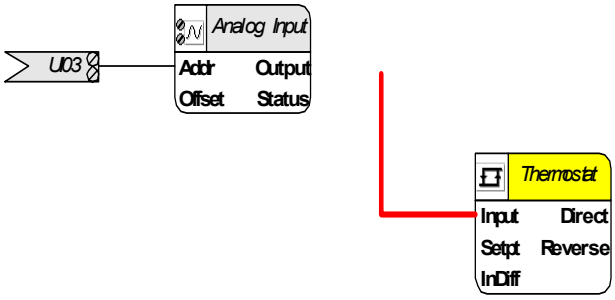
The following rules apply when connecting objects and tags in WP Tech:

- *Object inputs* must be connected to a single output, a constant, a variable reference tag, or an input tag.
- *Object outputs* can be connected to other object's inputs, output tags, or variable definition tags. Object outputs can be connected to as many other objects or tags as needed.
- *Hardware input tags* and *hardware output tags* can be connected to only one Addr (address) input or output of a single object.
- The output of an object cannot be directly connected to its own input. A wire extension must be used. For more information, see Wire Extension (page 35).

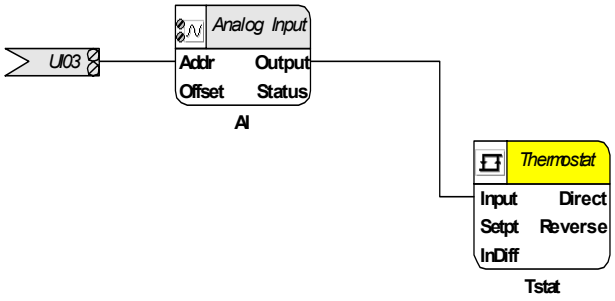
Two types of wire are used to link shapes together: built-in connection wires and the Wire Extension shape. In both cases, the wire becomes red whenever a connection error occurs, which means that the wire is either not connected, or is illegally connected.

Built-In Connection Wire

Control object shapes in WP Tech are specifically made to link to each other (and to input tags) by providing a built-in *connection wire* for each object *input*, plus a connection point for each object *output*. Any object's input is linked graphically by selecting it, using the mouse, and "pulling out and dropping" the built-in connection wire onto another object's output (or an output of a tag), as shown in Figure–2.11 and Figure–2.12. Output tags also have a single input with a connection wire that works in the identical manner (for connection to an output of a control object or a tag).

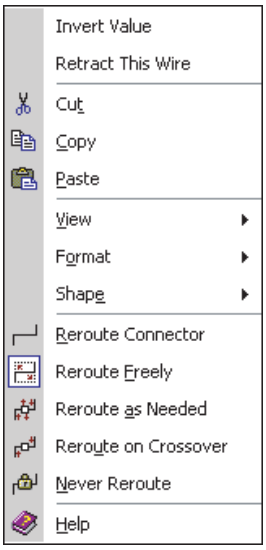


Figure–2.11 Selecting and Pulling Out a Control Object's Input.



Figure–2.12 Dropping the Connection Wire onto an Output.

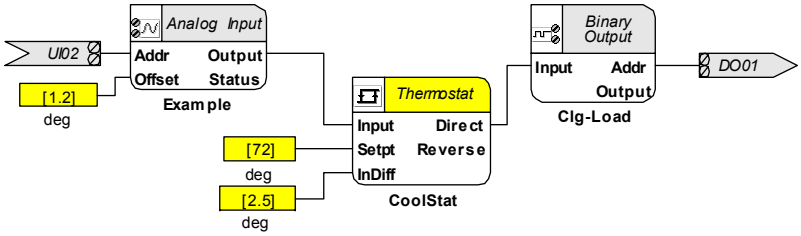
Note: When working with connection wires, a shortcut menu may be used to select one of five *reroute options* Figure-2.13. If you are already familiar with this menu in Visio, you should note that, while a single right-click on a wire causes this menu to appear in Visio, in WP Tech two left-clicks (not a double-click), then one right-click on the wire are needed to call up this menu. If the object and the wire are both selected, the command will not apply to the wire itself. This is necessary to ensure that you have selected the wire, and not the object. WP Tech treats the first left-click as a request to select *both the wire and the object* from which it was pulled. The second left-click prompts WP Tech to shift the focus to the wire itself, de-selecting the object. Then, with only the wire selected, a *right-click* reveals the wire's shortcut menu.



Figure–2.13 Right-Click Shortcut Menu for Connection Wires.

This method of linking control objects in WP Tech is called a *connection*. When the application drawing is compiled and downloaded into the controller, the controller's database stores a record of all the control objects, data from tags, and all the connections among them.

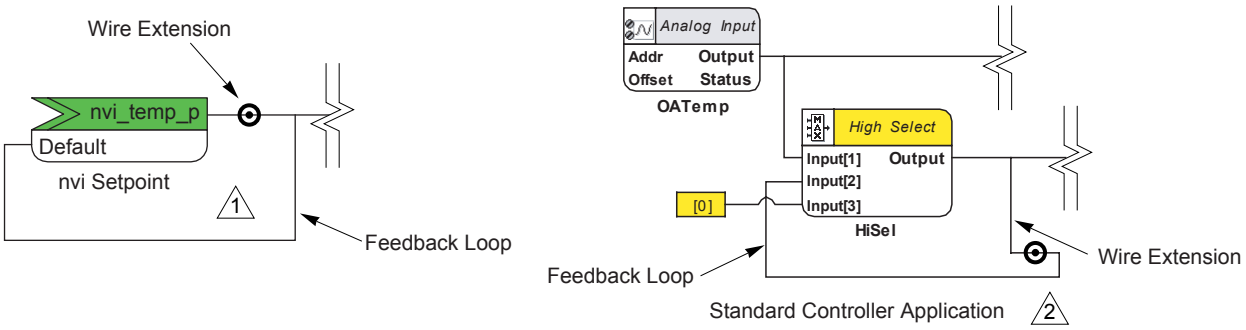
Control object discussions in this manual are illustrated using WP Tech shapes, as shown below Figure-2.14. This example represents the equivalent control logic shown in a previous example Figure-2.6.



Figure–2.14 Simple Three-Object Example as Seen in WP Tech.

Wire Extension

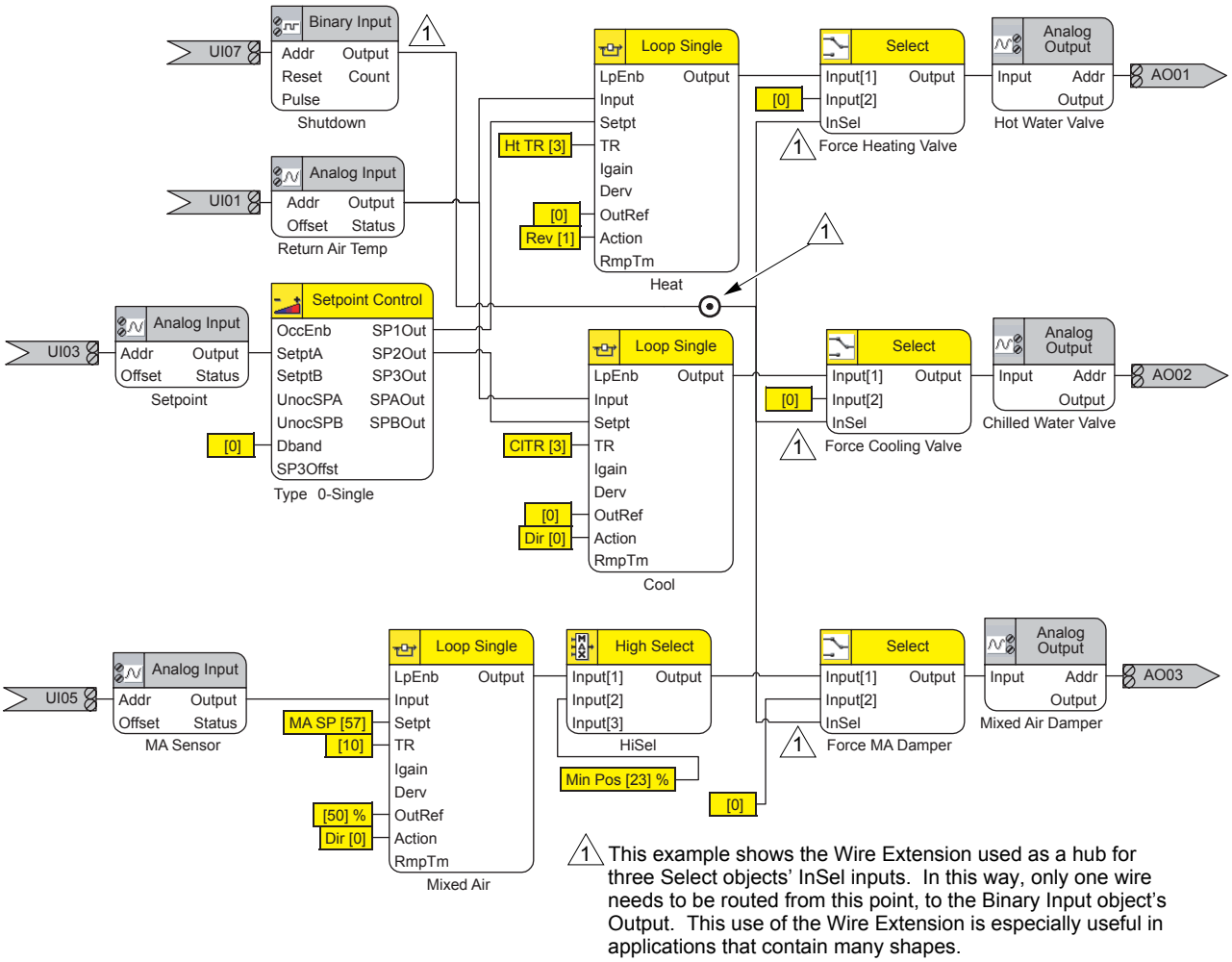
One of the Invensys shapes that appears on every stencil (except Annotations) is the Wire Extension shape. This shape may be copied onto a drawing and used to connect inputs and outputs in the same way built-in wires are used ("Built-In Connection Wire" on page 33). The Wire Extension shape, provided in all controllers, is unique in that it may also be used to create feedback loops. A feedback loop connects an output to an input on the same object.



1 Example shows an NVI object that receives a temperature value from a controller. A feedback loop has been created to continually replace the Default value with the latest valid value received from the controller. This value is maintained in the event communication with the controller is interrupted, so that operation can continue.

2 Example shows a simple method for monitoring an active value and maintaining the highest level obtained since the application download or a controller reset.

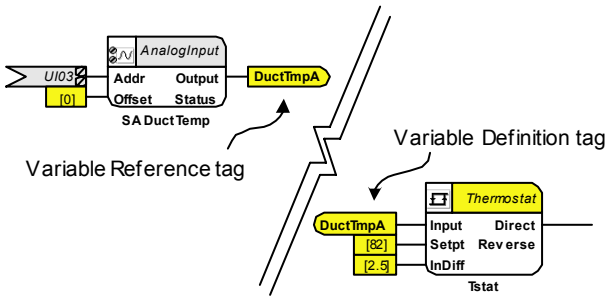
Figure–2.15 Wire Extension Shapes and Creation of Feedback Loops.



Figure–2.16 Wire Extension Shape Used to Connect Multiple Inputs to One Output.

Using Variable Definition and Reference Tags

An alternative to using connecting wires to connect object outputs to inputs is to use Variable Definition and Variable Reference tags. Found on every WP Tech stencil (except Annotations), these tags allow you to connect valid object outputs to valid object inputs, without the use of connecting wires. This can reduce the amount of wiring on a drawing page, thereby improving its readability. In addition, these tags can be used to connect a single output (Variable Definition Tag) to multiple inputs (Variable Reference Tags) on the same drawing. Variable Definition tags carry the output to one or more corresponding Variable Reference tag(s).



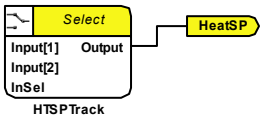
Figure–2.17 Variable Definition and Reference Tags.

To define and use Variable Definition and Reference tags:

Note: When creating multiple sets of variable tags, it is best to create one set at a time, to avoid connection errors.

1. Locate the object output that you want to use as the source.
2. Click and drag a Variable Definition tag from the stencil to an area on the drawing page near the chosen object output.
3. While the Definition tag is still selected, click and drag its connection wire onto the source output. The tag and its connection wire should remain active (selected).
4. Name the Definition tag as follows by right-clicking the Definition tag, then click **Rename**. Type a name in the text box and then click **OK**.

   In the example shown in Figure–2.18, the name of the Definition tag is HeatSP (heat setpoint). This tag will carry the output from the connected Select object to one or more corresponding HeatSP Reference tag(s) and their connected object input(s).



Figure–2.18 Named and Connected Variable Definition Tag.

5. Locate the object input that will serve as the destination for the Variable Definition tag you have just created.
6. Click and drag a Variable Reference tag from the stencil to an area on the drawing page near the destination object input.

Note:

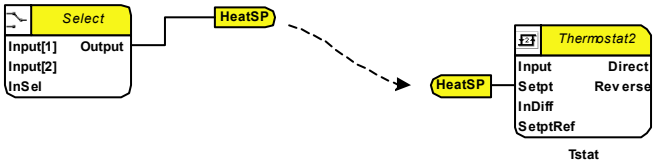
The Reference tag assumes the name of the last Definition tag that you placed on the drawing.

7. If the Reference tag is to be matched to an earlier Definition tag, name or rename the Reference tag by right-clicking the tag and click **Rename**. Open the **Select a defined variable name** drop-down list and select a name that corresponds exactly to the matching Definition tag. Click **OK**.

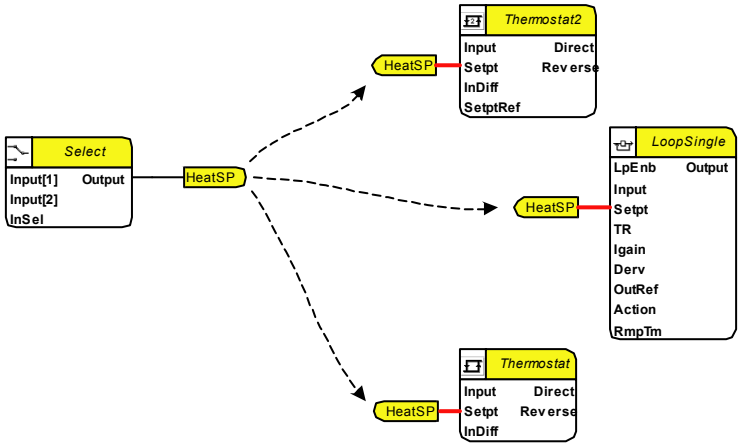
Tip: You can double-click to rename the tag.

8. Select the destination object input, then click and drag its connection wire onto the Variable Reference tag's connection point.

In the example shown in Figure–2.19, the Variable Definition and Reference tags named HeatSP are used to connect the Select object's output value to the Thermostat 2 object's Setpt input. Figure–2.20 illustrates how the same output may be connected to multiple inputs.



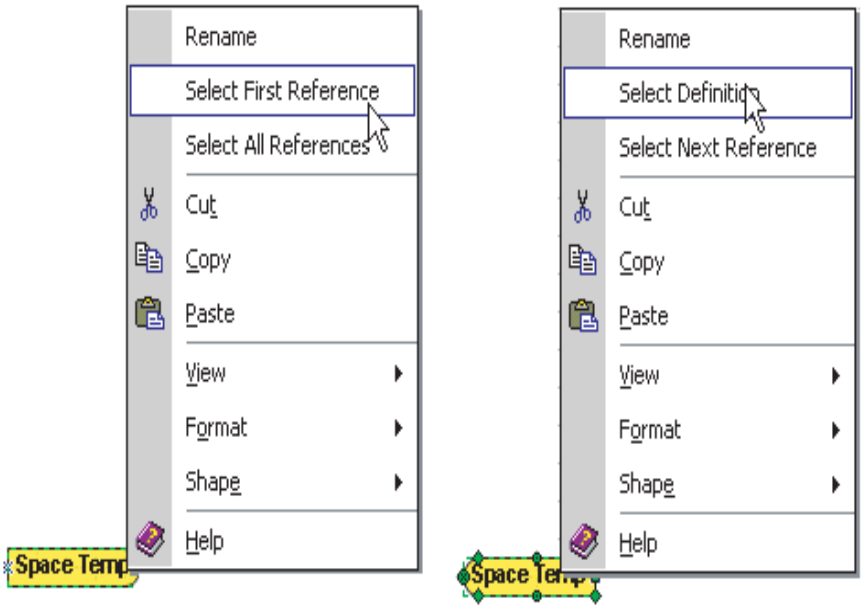
Figure–2.19 Output and Input Connected Using Variable Definition and Reference Tags.



Figure–2.20 Output Connected to Multiple Inputs Using Variable Definition and Reference Tags.

Variable Definition and Reference Tags

WP Tech provides shortcut menus to locate Variable Definition and Reference tags on a drawing. This is handy on complex applications where you might have more than one Reference tag connected to a Definition tag.



Figure–2.21 Variable Definition and Reference Tag Shortcut Menus.

To locate a Variable Definition or Reference tag:

1. Locate and click a related Variable Definition or Reference tag to select it.
2. Right-click the tag and on the shortcut menu, do one of the following:
   - If you selected the Definition tag, click either **Select First Reference** or **Select All References**. The tag(s) will be selected on the drawing.
   - If you selected the Reference tag, click either **Select Definition** or **Select Next Reference**. The appropriate tag will be selected on the drawing.

Variable Definition and Reference Tag Rules

- Variable Definition tags should be placed on the drawing before the Reference tag is placed and defined.
- Connections to Variables will be red until both Definition and Reference tags are properly connected.

Engineering Process Overview

WP Tech provides blank application templates for custom applications. Alternately, one of the Invensys pre-engineered applications can be used, either as is, or modified to meet specific requirements.

Custom Applications

A custom application can be built “from scratch” based on a blank application template. The template contains all the Invensys object stencils and utilities needed.

The blank application template can be sized to any of the following:

- A - 11.0”. x 8.5”
- B - 17.0” x 11.0”
- C - 22.0” x 17.0”
- D - 34.0” x 22.0”
- E - 44.0” x 34.0”
- A specified custom size

When a custom application is created, the Hardware Wizard is used to define the specific I/A Series MicroNet controller model and MicroNet sensor for the application. This creates all the available resource tags that can be used in the application, assembled in groups to the left, right, and below the drawing area. Input resource tags are to the left. Output resource tags are to the right. Mandatory SNVT objects appear below the drawing area.

During engineering, control objects and object tags are copied (as needed) from the object stencils onto the drawing area, while resource tags are moved as needed onto the drawing area. The necessary connections between objects (and tags) are defined by dragging wires from object inputs to outputs. At any time, an application statistics function allows a check on the amount of available memory left for additional control objects, corresponding to the target controller model. When the control logic is defined, the application can be compiled to a device image. Custom applications may be saved for reuse as “custom templates,” in the My Solutions folder located in the following default directory:
My Documents\WPT Documents\My Solutions Folder

The compile process aborts with error message(s) and generates a log file if errors are detected in the application drawing. The file provides a record of the most recent compile.

The error log file can be found in the same location as the application drawing. It will have the same name as the drawing, but can be identified by the “.err” extension.

All compile error messages must be investigated and corrected in the control logic drawing. Only after a successful compile (without errors) can the target controller be downloaded with a device image (hex file).

Note:

- The above also applies to any pre-engineered application that has been modified.
- Unresolved, or open, connections and objects that are not supported by the target controller are indicated in bold red and must be corrected before compiling the application.

Pre-engineered Applications

Pre-engineered templates are organized in subdirectories by the specific LONMARK profile or equipment type used, for example, Fan Coil Units, Heat Pumps, Roof Top Units, Unit Ventilators, or VAV controller. Each application template includes all the connected control objects and tags used in the application, and can be printed on B-sized (11" x 17") paper. These applications are tested and verified LONMARK-compliant for use in their intended I/A Series MicroNet controller models, and ready for downloading.

The Visio template for each Invensys pre-engineered application includes an embedded Word document for that application. The document provides a sequence of operation, materials list, wiring diagram, and control diagrams.

See "Pre-engineered Applications" section in Chapter 1 of this manual (page 15) for information on obtaining pre-engineered applications.

Downloading Applications

Once an application has been successfully compiled, it can be downloaded to the corresponding controller. Downloading is an online function, which means that the WP Tech workstation must be able to communicate with the controller over the LON. To allow this, the workstation must have an installed and working Echelon PC/LONTALK adapter with an assigned LON address that matches the domain and subnet of the network segment to which WP Tech is directly connected. This addressing is performed using WP Tech's Workstation Addressing Wizard.

Workstation Addressing Wizard

The Workstation Addressing Wizard is used to automatically or manually synchronize the WP Tech workstation's Echelon PC/LONTALK adapter to the domain and subnet of the LON network segment to which it is directly connected. This address synchronization is required by the Download, Upload, Monitor, and Browse Network functions of WP Tech.

Improper domain and subnet address setup of the WP Tech workstation can cause it to be unable to perform online functions such as Download, Upload, Monitor, and Browse Network:

- Downloading, uploading, or monitoring with an unsynchronized domain and subnet address causes WP Tech to indicate a "failure to communicate" error.
- Browsing with an unsynchronized domain and subnet address prevents WP Tech from properly displaying a complete list of all MicroNet controllers on the network.

Note:

- Before using the Workstation Addressing Wizard, the workstation must contain an installed, working LON adapter with the appropriate driver, and must be properly connected to the LON.
- If the workstation is being re-connected to a LON, and it has not been connected to any other LON in the intervening period, it does not have to be addressed again. However, workstation addresses can be overwritten at any time with a network management tool that is connected to the network. For this reason it is recommended that the Workstation Addressing Wizard be run each time, to ensure proper communications.
- If **Cancel** is clicked at any point while the Wizard is in use, the address reverts to the one that existed prior to starting the Wizard.
- Upon startup, WP Tech initializes the Echelon PC/LONTALK adapter with the address assigned by the Workstation Addressing Wizard.

The Wizard can be used to set the workstation address in one of three ways: to set the address automatically, manually, or to the factory default.

Setting the Workstation Address Automatically

Using the Workstation Addressing Wizard to automatically set the LON address is the recommended method. This is the simplest approach, and it eliminates any chance of error. To use this method, the Wizard is started and the **automatically synchronize** option is selected. Then the Neuron ID of a controller that shares the same LON network segment is entered, either by directly typing it into the text field, or by using the **Listen** button and pressing the controller's service pin. This completes the addressing process. At this point, communication between the workstation and the LON can be verified using the **Browse Network** button, and the **View Address** button can be clicked to view the Workstation Address property sheet.

Setting the Workstation Address Manually

The Workstation Addressing Wizard may also be used to manually set the LON address, by starting the Wizard and selecting the **manually set** option. This brings up a dialog box in which the user enters the Format, Domain ID, Subnet ID, and Node number. This completes the addressing process. At this point, communication between the workstation and the LON can be verified using the **Browse Network** button, and the **View Address** button can be clicked to view the Workstation Address property sheet.

Setting the Workstation Address to the Factory Default

For standalone use, the Workstation Addressing Wizard can be used to set the WP Tech workstation's LON address to the factory default. This is done by starting the Wizard and selecting the **factory default** option. By selecting this option, the Wizard assigns the factory default address to the workstation and the address verification dialog box appears. Communication between the workstation and the LON can then be verified using the **Browse Network** button, and the **View Address** button can be clicked to view the Workstation Address property sheet.

Note: Addressing conflicts may occur between the LON card used by WorkPlace Tech and third party tools such as LonMaker and I/A Series Niagara. The common assignment of node 127 in tool addressing can cause such a conflict. When this occurs monitor tags may display unexpected values or fail to update. If this occurs, it may be necessary to change the node address.

Realtime Application Checkout (Diagnostics)

Any application can be examined in real-time after it has been downloaded to a controller. This is done in WP Tech by using the application "Connect" function. The Connect function communicates to a selected I/A Series MicroNet controller and performs the following directly on its application's drawing page:

- Activates monitor tags. Monitor tags can be copied from object stencils, connected to outputs of objects or tags, and moved about and deleted/connected/reconnected where needed.
- Permits temporary writes to any fixed value input (constant tag, write to RAM).
- Permits writes to any Network Configuration Input value (NCI tag, write to NCIs).

Realtime checkout is typically done following any application download (whether a pre-engineered or a custom application) to test the basic operation of the application in the controller. This function is also useful when troubleshooting control logic problems or when calibrating sensor readings using offsets.

Control Object Categories

The two major categories of control object types are I/O point objects and functional objects. Each category contains subcategories of object types.

I/O Point Objects

I/O point objects include all object types used to interface to physical input/output (I/O) points on an I/A Series MicroNet controller, where "point" refers to a specific hardware I/O terminal. Subcategories of I/O point objects are *input* point objects and *output* point objects.

- An input point object monitors a controller's hardware input.
- An output point object activates a controller's hardware output.

In WP Tech, master shapes for all but one type of I/O point object are found on the object stencil "IO and Alarm Control". The exception is the Event Indicator object, found on the "Timer & Sequence Control" stencil.

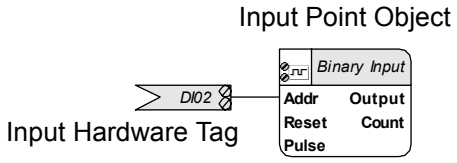
Input Point Objects

Each input point object has one object input reserved for assignment of a physical address. A physical address corresponds to a specific hardware input point on the MicroNet controller. The object processes the physical signal received at this input and (depending on its object type) produces a numerical output as an analog or digital value.

Input point objects include these types:

- Analog Input
- Binary Input
- DUI Expander
- Pressure Transducer (applies to VAV controllers only)
- Sensor Input (applies to MN 800 controller only)

In WP Tech, the shape for an input point object shows the physical address input (Addr) as the top input. This input can be connected to an available (and appropriate) input hardware tag.



Figure–2.22 Physical Address Assignment Using an Input Hardware Tag.

Note: A one-to-one rule is used when assigning a physical address to an input point object. (This means only one input point object can be connected to any input hardware tag.)

Output Point Objects

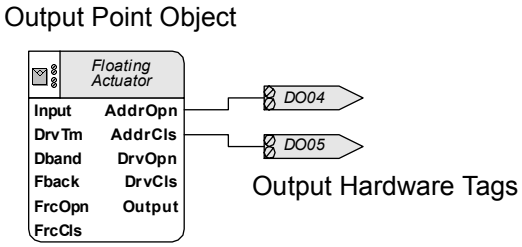
Each output point object has at least one object output reserved for assignment of a physical address. A physical address corresponds to a specific hardware output point on the MicroNet controller. The object processes received control logic value(s) and depending on its object type, activates/deactivates the assigned controller outputs.

Output point objects include these types:

- Analog Output
- Analog Output Priority
- Binary Output
- Event Indicator
- Fan Speed
- Floating Actuator
- Floating Actuator Priority
- Momentary Start / Stop
- PWM
- PWM Priority
- VAV Actuator (applies to VAV controllers only)

In WP Tech, the shape for an output point object shows the physical address (Addr) output(s) at the top of the outputs. Each Addr output can be connected to an available (and appropriate) output hardware tag.

Note: A one-to-one rule is used when assigning a physical address to an output of an output point object. (This means only one output hardware tag can be connected to any Address output.)



Figure–2.23 Physical Address Assignments Using Output Hardware Tags.

I/O Differences Among Hardware Platforms

The various controller platforms (MN 50, 100, 110, 130, 150, 200, 800, and VAV series) have different complements of physical I/O points. For example, MN 800 controllers have four 4 to 20 mA AO points, two of the three MN-VAV models (V2R and V3R) each have one AO point, and the MN 50 and 100 have no AO points. Note that an Analog Input object “compiles” in applications for all controllers above. However, it cannot be connected to any hardware tag in an application for an MN 50 or 100, therefore it has no real purpose (other than to allow application compatibility across hardware platforms).

Refer to “Understanding Programming Boundaries” section in Chapter 3 of this manual (page 51) for detailed listings of I/O point capacities for I/A Series MicroNet controller models.

Functional Objects

Functional objects include all control object types except I/O point objects. They do not correspond directly with any physical I/O points on an I/A Series MicroNet controller, but perform various routines used in different control logic applications. There are three general subcategories of functional objects as reflected in these WP Tech object stencils:

- Logic and Math Control
- Loop and Process Control
- Timer and Sequence Control

Two additional functional objects are found on the “IO and Alarm Control” stencils; the Analog Alarm object and Binary Alarm object.

Alarm Objects

The alarm objects can be used to detect alarm conditions and write a unique alarm message ID to the MicroNet controller’s local alarm buffer. The local alarm buffer can be reviewed at an attached MicroNet sensor with LCD display. Each alarm object features a programmable alarm delay time. There are two types of alarm objects:

- Analog Alarm
- Binary Alarm

Logic and Math Control Objects

These objects perform specific math operations of three main types:

- Logic objects act as digital “gates” using boolean math.
- Math objects apply an algebraic algorithm to analog values.
- Special-purpose objects perform a diverse variety of functions.

Most logic and math objects have three inputs and a single output. They are among the simpler of the functional objects. Control objects included in these categories are:

Logic Objects

- AND / AND
- AND / OR
- Clocked SR
- EXOR
- Latch
- OR / AND
- OR / OR
- SR Flip-Flop

Special-Purpose

- Compare
- Compare 2
- Count Down
- Count Up
- Curve Fit
- Demux Select

Math Objects

- Abs Sub / Div
- Add / Add
- Add / Div
- Average
- Enthalpy
- Filter
- MA Volume
- Mul / Add
- Sq Rt Mul / Div
- Sub / Add
- Sub / Div
- Sub / Mul
- Sub / Sub

Loop and Process Control Objects

These control objects perform various functions relating to direct digital control (DDC), HVAC control, or priority handling of data. They range from simple objects (Select object) to ones with complex control algorithms (Loop Sequenced object). Included object types are:

- Binary Encoder
- Control Override
- COV Priority
- High Select
- Interlock
- Limit
- Limit Thermostat
- Loop Sequenced
- Loop Single
- Low Select
- Priority Input (2)
- Priority Input (4)
- Priority Value Select
- Ramp (MN 800 only)
- Reset
- Select
- Sensor Input
- Setpoint Control
- Thermostat
- Thermostat 2

Timer and Sequence Control Objects

These control objects perform various time-based functions, including delays, minimum on or off periods, or output sequencing. Included object types are:

- Dual Delay
- Dual Minimum
- Event Indicator
- Interstage Delay (3)
- Interstage Delay (6)
- Interstage Delay (10)
- Minimum Off
- Minimum On
- Off Delay
- On Delay
- Sequence (3)
- Sequence (6)
- Sequence (10)
- Step Driver (MN 800 only)

Schedule Control Objects (MN 800)

These control objects, used only with the MN 800, perform various calendar/schedule based functions. Included object types are:

- Calendar
- OSS — Optimum Start Stop
- Schedule 7-Day

Migrating WP Tech 2.0 or 3.0 Projects into WP Tech 3.2

Projects created in WP Tech 2.0 and 3.0 may be migrated into WP Tech 3.2. To do so, first these earlier projects are copied into the WP Tech 3.2 Projects directory (X:\Program Files\Siebe\WIBs\WPTech\Projects), using standard Windows techniques. Then the projects and their applications are “recreated” so that WP Tech 3.2 recognizes them. This is accomplished by creating a new project, using the New Project dialog box. For detailed instructions on migrating earlier projects into WP Tech 3.2, refer to the *I/A Series WorkPlace Tech Tool 3.2 User’s Guide*, F-26987.

Note:

- When migrating WP Tech 2.0 projects into WP Tech 3.2, the Schedule stencil does not open with the application, nor does the Annotations stencil for Rev.3 and earlier applications. You can easily add these stencils by clicking the **Open Stencil** button on the tool bar and selecting the stencils from the Open Stencil dialog box.
- If an application created with WP Tech 2.0 is converted for use with a Rev. 3 controller, using the WP Tech 3.2 Hardware Wizard, the NCIs for heartbeat functions (i.e. nciSndHrtBt) will have connection tags. In this case, these tags are not functional and users should not connect to them.

Changing the Application Background

When an existing application is migrated into WP Tech 3.2, its background page still contains the “Siebe” logo and the company name, “Siebe Environmental Controls.” An older application drawing may be updated to the “Invensys” identity by first opening its background page and deleting the Siebe background. Then, the Invensys Background shape is dragged onto

the background page from the Annotations stencil until it snaps into place. For detailed instructions on changing the background page of an earlier project, refer to the *I/A Series WorkPlace Tech Tool 3.2 User's Guide*, F-26987.

Opening WP Tech 3.1 Projects in WP Tech 3.2

Projects created in WP Tech 3.1 can be opened directly in WP Tech 3.2. No special migration procedures are required.

Note: Opening a WP Tech 3.1 application in WP Tech 3.2 *for the first time* requires extensive processing. For very large applications, this may require up to 80 minutes to complete, although most conversions will take much less time. Because this processing occurs in the background, it may appear as though the system has become unresponsive. Please be patient. Once these applications have been opened and saved for the first time, they will open normally.

Migrating Projects into WP Tech 4.0

Migrating from WP Tech 2.0, 3.0, or 3.1

Projects created with WP Tech 2.0, 3.0, or 3.1 must be migrated to WPT 3.2 before migrating to WP Tech 4.0. Follow the procedures described above to migrate a project to WP Tech 3.2.

Migrating WP Tech 3.2 Projects into WP Tech 4.0

For detailed instructions on migrating earlier projects into WP Tech 4.0, refer to the *I/A Series WorkPlace Tech Tool 4.0 User's Guide*, F-27255.

Chapter 3
Understanding Programming Boundaries

This chapter explains programming boundaries when engineering a basic (standalone) I/A Series MicroNet controller application in WP Tech. Boundaries are device-specific and also logical in nature, and provide the framework in which control objects can be added, connected, and downloaded to the controller. Explained in later chapters are details on control objects and network variables.

Device-specific boundaries are determined by the selected model of MicroNet controller and MicroNet sensor. Typically, these boundaries are represented in a WP Tech application by *resource tags* for these items:

- Controller I/O Points (Hardware tags)
- MicroNet Sensor Attributes (Sensor tags)
- Other Resource Tags (Standard Controllers)

Another device-specific boundary is controller memory, which is consumed by control objects and, if an MN 800, by SNVT objects as well. The section "Controller Memory (RAM and EEPROM) (page 80)" explains this boundary.

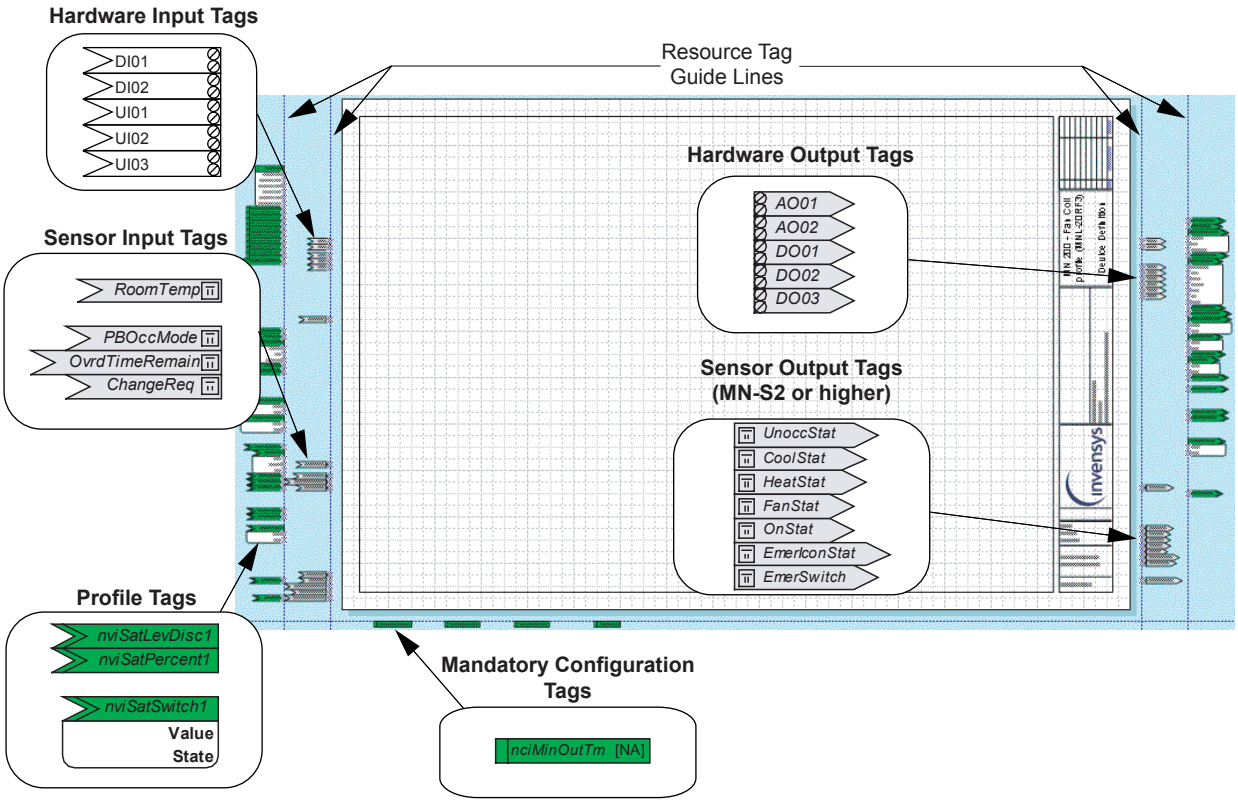
Logical boundaries vary with the number of Custom Objects (page 82) created, if any. A Custom object resides not in the controller but in WP Tech, and represents a logical grouping of controller-resident objects as a single object, providing "encapsulation". This modular approach is useful when creating an application with large numbers of objects, such as typical in an MN 800 application. Each Custom object adds an underlying Custom object definition page in the application drawing. This affects the appearance of the application in WP Tech, but not the actual application as it resides and operates in the controller.

Resource Tags In General

After running the Hardware Wizard in WP Tech or adding a new application to a project, resource tags are initially placed on “guide lines” in the blue “pasteboard” area outside the left and right side edges of the application drawing area (Figure–3.1). Resource tags include both *input* types (on the left side) and *output* types (on the right side).

All I/A Series MicroNet controllers have the following resource tags at the guide lines:

- Physical I/O points on the controller (hardware tags).
- MicroNet sensor functions (sensor tags).



Figure–3.1 Resource Tags Available in typical WP Tech LONMARK Application.

For standard controllers (MN 50, 100, 110, 130, 150, 200, or VAV series), the following resource tag types are included in addition to the tag types listed above.

- NVs (Network Variables) in the LONMARK profile (profile tags).
- Controller schedule and software clock functions (schedule tags), available if the controller has Rev.3 or higher firmware.

These resources are available for the application, and determine the programming *boundaries*. Unlike control objects, resource tags cannot be duplicated or erased within an application. Instead, each tag represents some controller or MN sensor function that may (or may not) be used in the application. Any resource tag is used by simply dragging it into the drawing area and connecting it as needed to control objects and other resource tags. Any resource tag not needed is simply left off of the drawing area.

If the application is for an MN 800, several default "SNVT objects" appear on the application drawing. These objects are like resource tags in a standard controller application (cannot be deleted or copied). They represent a small core set of NVs common to every MN 800. For more information, refer to "MicroNet MN 800 Controllers (page 548)."

Resource Tag Rules

These rules apply to resource tags in any type of WP Tech application (I/A Series MicroNet controller application).

- Resource tags cannot be deleted or copied.
- If not needed, a resource tag should be left off the drawing, on the appropriate guide line.
- Resource tags can be used only on the top page (Controller Definition page) of any application. Note an application without Custom objects has only this one page. However, if Custom objects are added, each one adds an underlying Custom object definition page. Resource tags cannot be used on these definition pages.

Controller Type Considerations

The following table shows the resource tags present when engineering one of the three types of I/A Series MicroNet controllers programmable in WP Tech. Note that early MicroNet standard controllers (pre-Rev.3 MN 100, 200, VAV) do not have a controller-resident schedule and associated Schedule tags.

Table–3.1 Resource Tags Present When Programming I/A Series MicroNet Controllers in WP Tech.

Resource Tag Type	MicroNet LonWorks (MN 800)	MicroNet LonMark	
		MN 50, 100, 110, 130, 150, 200, VAV (Rev.3 or later)	MN 100, 200, VAV (Pre-Rev.3)
Hardware Input	Yes	Yes	Yes
Hardware Output	Yes	Yes	Yes
Sensor Input	If MN-Sx	If MN-Sx	If MN-Sx
Sensor Output	If MN-S2 or higher	If MN-S2 or higher	If MN-S2 or higher
Profile: NCI Input	No, NCI objects instead	Yes, per profile	Yes, per profile
Profile: NVI Input	No, NVI objects instead	Yes, per profile	Yes, per profile
Profile: NVO Output	No, NVO objects instead	Yes, per profile	Yes, per profile
Schedule Input	Yes[a]	Yes	No
Schedule Output	Yes[a]	Yes	No

a.As well as Schedule tags, the MN 800 has additional resources in the form of Calendar and Schedule 7-Day objects.

More on Rev.3 or Later vs. Pre-Rev.3 Standard Controllers

In addition to built-in schedule functions (schedule tags), standard controllers with Rev.3 or higher firmware have other functionality, summarized here. These changes are also noted in context in further sections of this chapter.

Table–3.2 Resource Tag Differences, Rev.3 or Later vs. pre-Rev.3 Controllers.

Rev. 3 or Later Firmware	pre-Rev.3 Firmware (MNL-10Rx1, -20Rx1, -VxR1)
Up to 26 total sensor tags (MN-S5HT)	Up to 23 total sensor tags (MN-S5HT)
NVI tags can be individually excluded from nciRcvHrtBt function.	The nciRcvHrtBt function applies to all NVI tags
nviTimeStamp, nvoTimeStamp functions dedicated for controller clock and schedule use.	nviTimeStamp, nvoTimeStamp tags available to the application

Controller I/O Points

Each I/A Series MicroNet controller model has a fixed number of physical inputs and outputs, collectively known as I/O points. I/O points include controller inputs and controller outputs. Typically, a control application is engineered to use most (if not all) of the available I/O points on a controller.

Controller Inputs

A controller input is used to physically monitor a condition or state in the controlled environment (for example, a humidity level or an equipment status). Monitoring devices include sensors and contacts (switches). Wired to controller inputs, these devices are the “eyes and ears” of the application. Abbreviations and types of common controller inputs are:

- DI - Digital Input
- UI - Universal Input

Controller Outputs

A controller output is used to send a physical signal to a controlled device. Controlled devices often impact the measured environment, and include equipment relays and contactors, and actuators to position valves and dampers. Wired to controller outputs, these devices are the “arms and legs” of the application. Abbreviations and types of controller outputs are:

- AO - Analog Output
- DO - Digital Contact Output
- TO - Triac Output (MNL-V3RVx, MNL-11RFx, and MNL-13RFx models only)

Integral I/O Points

MicroNet VAV controllers provide additional “integral” I/O points, namely a built-in pressure transducer for measuring velocity pressure and (for most models) a built-in actuator for positioning a VAV terminal box damper. These I/O points require mechanical vs. wiring connections.

- Pressure (Input) - Integral VAV Pressure Transducer
- Actuator (Output) - Integral VAV Damper Actuator (not MNL-V3Rx)

I/O Point Capacities (by Controller Model)

Numbers of I/O points for each I/A Series MicroNet controller model are listed below (Figure-3.3), including the integral I/O points for the three VAV models.

Table–3.3 Numbers of I/O Points on I/A Series MicroNet Controller Models.

MicroNet Controller Model (Platform Abbreviation)	Inputs			Outputs				
	DI	UI	Pressure	AO	DO	TO	Actuator	Relay
MNL-5Rxx (MN 50 Controller)	1	1	—	—	3	—	—	—
MNL-10Rxx (MN 100 Controller)	1	2	—	—	4	—	—	—
MNL-11RFx (MN 110 Controller)	—	3	—	—	—	4	—	1[a]
MNL-13RFx (MN 130 Controller)	—	3	—	—	—	4	—	3[a]
MNL-15Rxx (MN 150 Controller)	—	3	—	2	2	—	—	—
MNL-20Rxx (MN 200 Controller)	2	3	—	2	6	—	—	—
MNL-V1RVx (VAV, Integral Actuator)	1	1	1	0	0	—	1	—
MNL-V2RVx (VAV, Integral Actuator, and extended I/O)	1	1	1	1	3	—	1	—
MNL-V3RVx (VAV, no Integral Actuator, and extended I/O)	1	1	1	1	3	2	—	—
MNL-800 series (MN 800 Controller)[b]	—	8	—	4	8	—	—	—

a.The MN 110 and MN 130 feature high-voltage (mains-rated) relay outputs capable of switching up to 3 A at 240 Vac.
b.The MN 800 has the same wiring backplane and number and types of I/O points as the MICROZONE II controller, a NETWORK 8000® device.

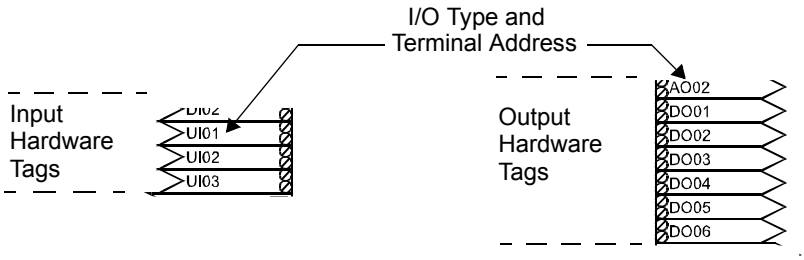
WP Tech Representations

In an application's control logic drawing, WP Tech represents each individual I/O point of a controller with a Hardware I/O Tag. Each tag depicts two screw terminals. These tags are generated (along with other resource tags) when running the Hardware Wizard and identifying a particular controller model.

The two types of hardware I/O tags are:

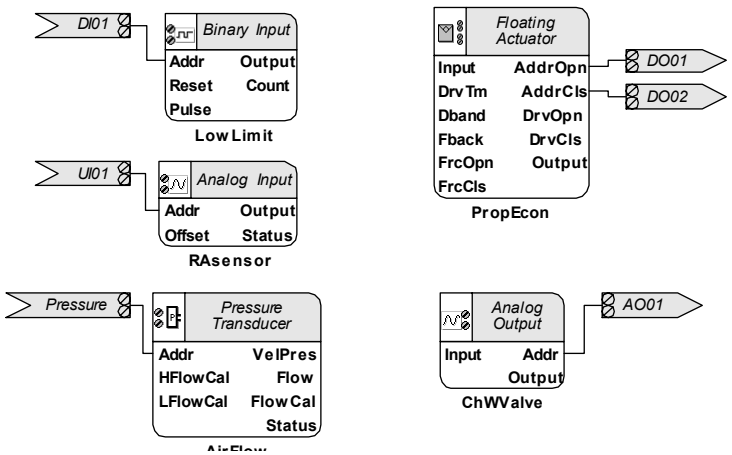
- Input Hardware Tags, which represent controller inputs.
- Output Hardware Tags, which represent controller outputs.

Each tag lists the I/O *type* and a specific terminal *address*. For example, any MN 200-based application has (among others) three UI-type input and two AO-type output hardware tags. These tags correspond to the universal inputs (UI1, UI2, and UI3) and analog outputs (AO1 and AO2) on the MN 200 controller. See the example below (Figure-3.2).



Figure–3.2 Some Example Hardware I/O Tags in a MN 200-based Application.

Hardware I/O tags can be moved about as needed onto the application drawing page and connected to I/O point objects (Figure–3.3). Input hardware tags connect to input point objects (Analog Input, Binary Input, and Pressure Transducer). Output hardware tags connect to output point objects.



Figure–3.3 Hardware I/O Tags Connected to I/O Point Objects.

Note: Hardware tags should be connected only to applicable Physical Address (Addr) inputs or outputs of I/O point object types. (A connection error results if a hardware I/O tag is connected to any other type of object input or output.)

MicroNet Sensor Attributes

A MicroNet standard controller or MN 800 controller can have a single MicroNet digital wall sensor. The sensor communicates digitally to the controller via an “S-Link” connection, and does not expend a traditional I/O point.

All MN sensor models provide a room temperature sensing value. MN-SxHT models also provide a room relative humidity value.The S2xx through S5xx models allow the user to control different functions using one or more buttons and an LCD screen. For example, an MN-S4 sensor-user can start a timed occupancy override, adjust one or more setpoints, and select a fan or HVAC operation mode. Each of these functions is an MN sensor attribute.

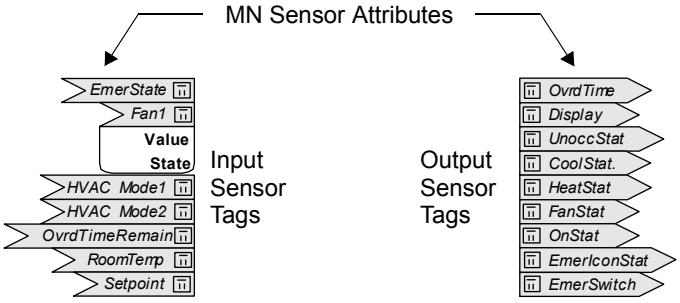
WP Tech Representations

In an application’s control logic drawing, WP Tech represents attributes of an MN sensor with individual *Sensor tags*. Each tag depicts a small stylized sensor (square box with three lines). Sensor tags are generated as a result of running the Hardware Wizard and identifying a particular MN sensor model and sensor-related options. Depending on the MN sensor model and options selected, from 1 to 26 sensor tags are generated. (No sensor tags are generated if the MN sensor is *none*.)

The two general categories of sensor tags are listed below.

- Input sensor tags represent user selections from the sensor (setpoint adjustment, mode selection, occupancy override) and the value of the integral room temperature and relative humidity sensors.
- Output sensor tags are generally used to control LCD screen features (MN sensor models S3xx, S4xx, S4xx-FCS, or S5xx only).

Sensor tags are labeled by the type of sensor attribute in the Hardware Wizard. For example, an application with an MN-S5 includes three sensor tags related to the Emergency (Heat) key. These sensor tags are EmerState, EmerSwitch, and EmerIconStat, as shown below (Figure-3.4).

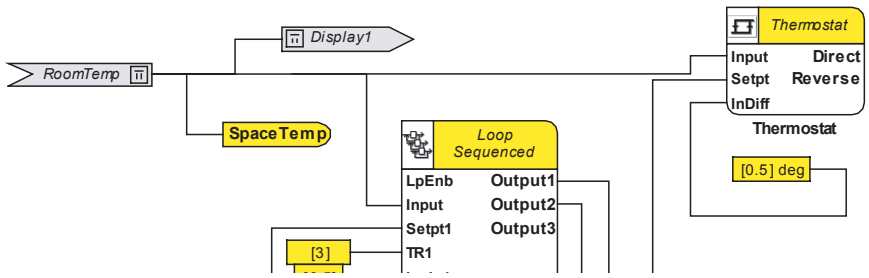


Figure–3.4 Some Example Sensor Tags in an Application with an MN-S5 Sensor.

Sensor tags can be moved about as needed onto the application drawing page and connected to control objects and other tags. Regardless of the MN sensor model and options selected, the resulting sensor tags can be thought of as “mini-objects”, whose use in an application establishes how the controller’s MN sensor operates.

For example, an MN-S4 sensor connected to an unprogrammed controller does not display temperature, nor does it provide access to other functions (such as setpoints or modes). This sort of sensor behavior must be engineered in the application by using and connecting the sensor tags generated by the Hardware Wizard.

Figure-3.5 shows how the sensor tag RoomTemp has been connected to two control objects, a variable definition tag, and another sensor tag (Display1) in an application that uses a MN sensor model with LCD screen (MN-S3xx, S4xx, S4xx-FCS, or S5xx).



Figure–3.5 Example Sensor Tags Used in an Application.

For details on each of the 26 types of sensor tags, refer to the next section, “S-Link Sensor (Sensor Tags)”. For details on running the Hardware Wizard in WP Tech, refer to the *WorkPlace Tech Tool 4.0 User’s Guide*, F-27255.

S-Link Sensor (Sensor Tags)

Usage: An S-Link Sensor object is represented in WP Tech as a collection of 1 to 26 resource “sensor tags” (versus a single object shape on an object stencil). WP Tech generates these sensor tags as a result of running the Hardware Wizard within an application. The number of available sensor tags in an application depends on the sensor model selected and other model-specific options enabled when running the Hardware Wizard.

Each sensor tag has either one to two outputs or one input for control logic connections.

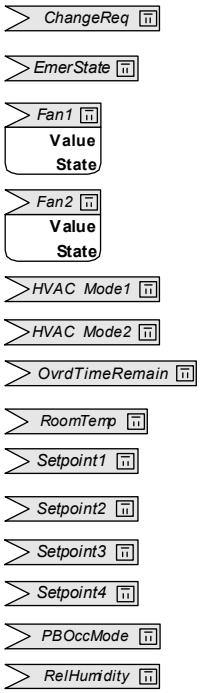
- An *input* sensor tag has one or two *outputs* that produce a sensor value or a representation of a sensor-user action, such as the measured room temperature, a user-adjusted setpoint, or an occupancy override initiated from a sensor push-button. All MN sensor models have at least one input sensor tag (RoomTemp).
- An *output* sensor tag has an *input* that provides access to a particular display feature (S3xx, S4xx, S4xx-FCS, S5xx) or to the override time for an S2xx, S3xx, S4xx, S5xx (MN 800 and Rev.3 or higher standard controllers).

The behavior of a MicroNet sensor is determined by how the sensor tags are connected to the application control logic (which in turn is compiled, downloaded, and *stored in the controller*).

WP Tech Representation (All 26 Sensor Tags)

Input Sensor Tags

(Name / Output Description / Applicable MN-Sxxx Models)



ChangeReq / (Change Request) / All but S1xx, S2xx (Rev.3 and later or 800 only)

EmerState / Emergency Switch Condition (Request) / S5xx only

Fan1State / Fan 1 State / S4xx, S4xx-FCS, or S5xx option

Fan1Value / Fan 1 Speed / S4xx, S4xx-FCS, or S5xx option

Fan2State / Fan 2 State / S4xx or S5xx option

Fan2Value / Fan 2 Speed / S4xx or S5xx option

HVAC Mode1 / HVAC Mode 1 Command / S4xx or S5xx option

HVAC Mode2 / HVAC Mode 2 Command / S4xx or S5xx option

OvrdTimeRemain / Override Minutes Left / All but S1xx, S4xx-FCS

RoomTemp / Sensor Temperature Value / All MN-Sxxx

Setpoint1 / Setpoint 1 Value / S3xx, S4xx, S4xx-FCS, or S5xx option

Setpoint2 / Setpoint 2 Value / S4xx or S5xx option

Setpoint3 / Setpoint 3 Value / S4xx or S5xx option

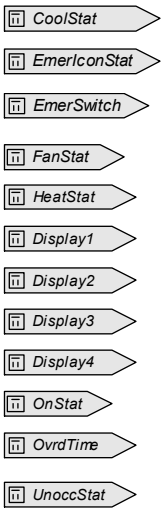
Setpoint4 / Setpoint 4 Value / S4xx or S5xx option

PBOccMode / Push-Button Override / All but S1xx, S4xx-FCS

RelHumidity / Sensor Relative Humidity Value/All MN-SxHT

Output Sensor Tags

(Name / Input Description / Applicable MN-Sx Models)



CoolStat / Cooling Mode Status LCD Icon / S4, S4-FCS, or S5

EmerIconStat / Emergency Heat (Request) / S5 only

EmerSwitch / Emergency Key Control (Request) / S5 (Rev.3 or later and 800 only)

FanStat / Fan Status LCD Icon / S4, S4-FCS, or S5 only

HeatStat / Heating Mode Status LCD Icon / S4, S4-FCS, or S5

Display1 / Default LCD Screen Value / S3, S4, S4-FCS or S5

Display2 / Second LCD Screen Value / S4 or S5 only

Display3 / Third LCD Screen Value / S4 or S5 only

Display4 / Fourth LCD Screen Value / S4 or S5 only

OnStat / ON Status LCD Icon / S4 or S5 / (AUTO icon if S4-FCS)

OvrdTime / (Rev.3 or later and 800 only) Override Duration (min.) / All but S1, S4-FCS

UnoccStat / Unoccupied Status LCD Icon / S4, S4-FCS, or S5

S-Link Sensor (Sensor Tags) (Continued)

Device Support and Memory Requirements: Using the Hardware Wizard in WP Tech, one S-Link Sensor object (represented with sensor tags) can be created in these I/A Series MicroNet controllers for application support of one MN-S1xx through S5xx sensor model:

MNL-5Rxx, -10Rxx, -11RFx, -13RFx, -15Rxx, -20Rxx
(MN 50, 100, 110, 130, 150, 200 series)
MNL-V1Rx, -V2Rx, -V3Rx (VAV series)
ENCL-MZ800-xxx, MNL-800-101
(MN 800 series)

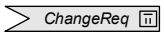
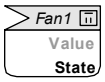
Note: Controller memory (RAM and EEPROM) is pre-allocated for this object.

Notes:

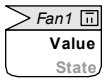
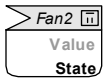
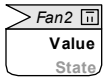
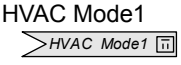
- The MN-S4xx-FCS model requires a Rev.3 or later MicroNet standard controller or an MN 800.
- Sensor tags may be renamed in WP Tech without affecting operation. The sensor tag names listed above reflect the default names generated by the Hardware Wizard.
- The input tag “ChangeReq” and output tags “OvrdTime and “EmerSwitch” are available only when programming Rev.3 or later and MN 800 controllers.

Sensor Tags

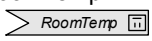
Table–3.4 Input Sensor Tags - S-Link Sensor Object.

Applies to MN-Sx	WP Tech Appearance (Default)	Output Class / Description	Default	Valid Values
All but S1xx or S2xx	ChangeReq 	Class: Analog - Momentarily outputs a value when an operator accesses (views or changes) the setpoint, mode, or fan setting of an MN-S3xx, MN-S4xx, MN-S4xx-FCS, or MN-S5xx sensor. The output value corresponds to the value of the accessed setting. If an operator accesses multiple settings in succession, the output equals the sum of the values corresponding to the accessed settings. (Available only if controller is either an MN 800 or a standard controller with Rev.3 or higher firmware)	0 (NA following a reset)	SP1 (1) SP2 (2) SP3 (4) SP4 (8) MODE1 (64) MODE2 (128) FAN1 (256) FAN2 (512)
S5xx only	EmerState 	Class: Digital - Outputs ON to indicate either condition: • An Emergency Switch request was received from the MN-S5xx sensor (Emergency Heat Key was pressed). • An ON value was received (in the application) at either the output sensor tag EmerIconStat or EmerSwitch. Alternate presses of the Emergency Heat Key toggle the EmerState output OFF and ON (providing that the EmerIconStat tag, if used, has an input of OFF or NA).	OFF following a download.	OFF (0.0) ON (100.0)
S4xx, S4xx-FCS, or S5xx	Fan1State 	Class: Digital - Outputs an ON (100.0) if any Fan1 action except AUTO is entered from the MN-S4xx or S5xx sensor. Selection of AUTO sets the output to OFF (0.0).	Configured in the Fan1 tab of the Hdw.Wiz.	ON (100.0) OFF (0.0)

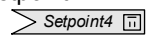
Table–3.4 Input Sensor Tags - S-Link Sensor Object. (Continued)

Applies to MN-Sx	WP Tech Appearance (Default)	Output Class / Description	Default	Valid Values
S4xx, S4xx-FCS, or S5xx	Fan1Value 	Class: Analog - Outputs a numerical value corresponding to a Fan1 action entered from the MN-S4xx, S4xx-FCS, or S5xx sensor. Available fan actions depend on the "fan type" defined in the Fan1 tab of the Hardware Wizard. Fan types that include an "Off" action are available only if the controller is an MN 800 or a standard controller with Rev.3 or later firmware. Available fan types: • Auto/On • 3 Speed/Auto • 2 Speed/Auto • 3 Speed • 2 Speed/ • Auto/On/Off • 3 Speed/Auto/Off • 2 Speed/Auto/Off • 3 Speed/Off • 2 Speed/Off For an MN-S4xx or S5xx, Fan1 fan type determines which LCD icons are shown for a fan-action after a *single press* on the sensor's Fan Key. The MN-S4xx-FCS has dedicated fan speed keys (High, Med., Low) plus a fan On/Off key. The default output value (configured in the Hardware Wizard) is only active following a controller download until a Fan1 action is entered from the MN sensor.	Configured in the Fan1 tab of the Hardware Wizard. (Any valid value for the selected fan type.)	Auto NA ON NA OFF (0.0) (3 Speed): LOW (33.0) MED (66.0) HIGH (100.0) (2 Speed): LOW (33.0) HIGH (100.0)
S4xx or S5xx	Fan2State 	Class: Digital - Outputs an ON (100.0) if any Fan2 action except AUTO is entered from the MN-S4xx or S5xx sensor. Selection of AUTO sets the output to OFF (0.0).	Configured in the Fan2 tab of the Hdw.Wiz.	OFF (0.0) ON (100.0)
S4xx or S5xx	Fan2Value 	Class: Analog - Outputs a numerical value corresponding to a Fan2 action entered from the MN-S4xx or S5xx sensor. Available fan actions depend on the "fan type" defined in the Fan2 tab of the Hardware Wizard. Fan types that include an "Off" action are available only if the controller is an MN 800 or a standard controller with Rev.3 or later firmware. Available fan types are: • Auto/On • 3 Speed/Auto • 2 Speed/Auto • 3 Speed • 2 Speed • Auto/On/Off • 3 Speed/Auto/Off • 2 Speed/Auto/Off • 3 Speed/Off • 2 Speed/Off Fan2 fan type determines which LCD icons are shown for a fan-action after *two presses* on the sensor's Fan Key. The default output value (configured in the Hardware Wizard) is only active following a controller download until a Fan2 action is entered from the MN sensor.	Configured in the Fan2 tab of the Hardware Wizard. (Any valid value for the selected fan type.)	Auto NA ON NA OFF (0.0) (3 Speed): LOW (33.0) MED (66.0) HIGH (100.0) (2 Speed): LOW (33.0) HIGH (100.0)
S4xxx or S5xx	HVAC Mode1 	Class: Analog - Provides a numerical output that corresponds to an HVAC Mode1 selection made at the MN-S4xx or S5xx sensor. The Mode1 option tab in the Hardware Wizard allows sensor selection of any or all of the following modes: • Heat • Cool • Auto • Off Modes made available determine which LCD icons are shown after a *single press* on the sensor's Mode Key. The default output value (configured in the Hardware Wizard) is only active following a controller download until a Mode1 selection is entered from the MN sensor.	Configured in the Mode1 tab of the Hardware Wizard. (Any valid value from the enabled modes.)	0.0 (AUTO) 1.0 (HEAT) 3.0 (COOL) 6.0 (OFF)

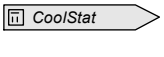
Table–3.4 Input Sensor Tags - S-Link Sensor Object. (Continued)

Applies to MN-Sx	WP Tech Appearance (Default)	Output Class / Description	Default	Valid Values
S4xx or S5xx	HVAC Mode2 	Class: Analog - Provides a numerical output that corresponds to an HVAC Mode2 selection made at the MN-S4xx or S5xx sensor. The Mode2 option tab in the Hardware Wizard allows sensor selection of any or all of the following modes: • Heat • Cool • Auto • Off Modes made available determine which LCD icons are shown after *two presses* on the sensor's Mode Key. The default output value (configured in the Hardware Wizard) is only active following a controller download until a Mode2 selection is entered from the MN sensor.	Configured in the Mode2 tab of the Hardware Wizard. (Any valid value from the enabled modes.)	0.0 (AUTO) 1.0 (HEAT) 3.0 (COOL) 6.0 (OFF)
S2xx, S3xx, S4xx, S5xx	OvrdTimeRemain 	Class: Analog - Indicates the remaining time, in minutes, of a timed override initiated from the sensor's Override Key. An override can allow the controller's unoccupied mode to change to occupied mode for a predefined period in minutes. Depending on the controller firmware level, the duration of a sensor-override is defined in this manner: • MN 800 or standard controller with Rev.3 or higher firmware - The value present at the output sensor tag "OvrdTime" (0 to 16383 minutes). • pre-Rev.3 firmware - A value from 0 to 120 minutes defined in the Hardware Wizard (no drawing visibility). After an override is initiated, the value counts down each minute towards zero (0), when the override expires. In the last 5 minutes of an override, the sensor's override LED flashes. After an expired override or an override canceled from the MN sensor, the output remains at zero (0).	(0) following a download or a power cycle.	0 (No Override) or 1 to 16383 or 1 to 120 minutes (Active Override.) Note: A value of 16383 indicates an active "forever" override, which does NOT count down.
All models	RoomTemp 	Class: Analog - Outputs the room temperature value as measured by the integral MN-Sxxx temperature sensor. This value is in either English (°F) or Metric (°C) numerical units (defined in the Hardware Wizard). If a sensor fault occurs, the output becomes not active (NA). Typically, the output value is used within the control application for space temperature control. In addition, any sensor model with LCD screen (MN-S3xx, S4xx, S4xx-FCS, or S5xx) requires this output to be connected to one of the available sensor Display tags if the room temperature is needed for display at the sensor. For the MN-S4xx or S5xx sensors that have four available display screens, the RoomTemp output is typically connected to the *first* Display sensor tag (Display1), which acts as the sensor's "default" display.	—	32 to 122°F (0.0 to 50.0°C)
SxHT, S4HT-FCS	RelHumidity 	Class: Analog - Outputs the room relative humidity values as measured by the integral MN-SxHT or MN-S4HT-FCS relative humidity sensor. If a sensor fault occurs, the output becomes not active (NA). Typically, the output value is used within the control application for space relative humidity control. In addition, any sensor model with LCD screen (MN-S3HT, S4HT, S4HT-FCS, or S5HT) requires this output to be connected to one of the available sensor Display tags if room relative humidity is needed for display at the sensor.	—	5 to 95% RH

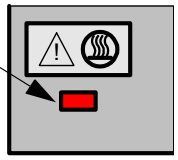
Table–3.4 Input Sensor Tags - S-Link Sensor Object. (Continued)

Applies to MN-Sx	WP Tech Appearance (Default)	Output Class / Description	Default	Valid Values
S3xx, S4xx, S4xx-FCS, or S5xx	Setpoint1  * Can be renamed & reordered in the Hardware Wizard.	Class: Analog - Outputs an analog value accessed at the MN-S4xx or S5xx sensor using the Setpoint Key (*single press*) and adjusted using the Up/Down Key. If an MN-S3xx or S4xx-FCS sensor, this analog value is accessed and adjusted using only the Up/Down Key. The Hardware Wizard provides a valid range of values (Min and Max) for adjustment from the sensor, and an initial value (Init) for this setpoint. The Hardware Wizard also allows various LCD screen features associated with displaying this setpoint value, including unit icons,e.g.; °F, °C, %, (and if an S4xx, S4xx-FCS, or S5xx, others such as Cool, Heat, Unoccupied). Setup in the Hardware Wizard also determines if this setpoint value is formatted for display at the sensor in tenths or in whole numbers.	Min, Max, and Init values are set using the Hardware Wizard. Defaults: 68ºF/18ºC (Min) 76ºF/26ºC (Max) 72ºF/22ºC (Init)	-99 to 999 is the display range of an MN-S3xx, S4xx, S4xx-FCS, or S5xx sensor.
S4xx or S5xx	Setpoint2  * Can be renamed & reordered in the Hardware Wizard.	Class: Analog - If created, outputs an analog value accessed at the sensor using the Setpoint Key (*second press*) and adjusted using the Up/Down Key. The Hardware Wizard provides the same (but separate) parameters as for Setpoint1 for the range of value, initial value, and display features for this value.	Same as for Setpoint1	-99 to 999 is the display range of an MN-S4 or S5 sensor.
S4xx or S5xx	Setpoint3  * Can be renamed & reordered in the Hardware Wizard.	Class: Analog - If created, outputs an analog value accessed at the sensor using the Setpoint Key (*third press*) and adjusted using the Up/Down Key. The Hardware Wizard provides the same (but separate) parameters as for Setpoint1 for the range of value, initial value, and display features for this value.	Same as for Setpoint1	-99 to 999 is the display range of an MN-S4 or S5 sensor.
S4xx or S5xx	Setpoint4  * Can be renamed & reordered in the Hardware Wizard.	Class: Analog - If created, outputs an analog value accessed at the sensor using the Setpoint Key (*fourth press*) and adjusted using the Up/Down Key. The Hardware Wizard provides the same (but separate) parameters as for Setpoint1 for the range of value, initial value, and display features for this value.	Same as for Setpoint1	-99 to 999 is the display range of an MN-S4 or S5 sensor.
S2xx, S3xx, S4xx or S5xx	PBOccMode 	Class: Digital - Outputs an ON during any active occupancy override (initiated from the MN sensor Override Key). Remains OFF following an expired override or an override canceled from the MN sensor.	(0) following a download or a reset.	OFF (0.0) ON (100.0)

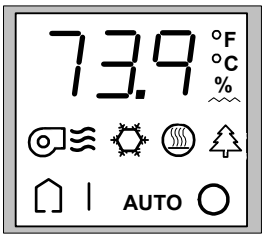
Table–3.5 Output Sensor Tags - S-Link Sensor Object.

Applies to MN-Sx	WP Tech Appearance (Default)	Output Class / Description	LCD Screen or Sensor Feature Controlled
S4xx, S4xx-FCS or S5xx	CoolStat 	Class: Digital - Allows the Cool icon to appear on the “default” screen of the MN-S4xx, S4xx-FCS, or S5xx LCD screen. The Cool icon appears when the input value is ON (value over zero). Otherwise, the Cool icon is not visible if the input value is OFF (zero or less) or NA.	

Table–3.5 Output Sensor Tags - S-Link Sensor Object. (Continued)

Applies to MN-Sx	WP Tech Appearance (Default)	Output Class / Description	LCD Screen or Sensor Feature Controlled
S5xx only	EmerIconStat EmerIconStat	Class: Digital - Controls Emergency (Heat) Key usage on an MN-S5xx sensor and affects the EmerState tag status, reflected by the Emergency (Heat) LED on the sensor. • An input value of ON (value over zero) lights the Emergency Heat LED *and* sets the output of the input sensor tag EmerState to ON (100.0). As long as the input is held ON, the sensor-user *cannot* toggle or turn OFF the Emergency State by pressing the Emergency key (the LED momentarily turns OFF but back ON again). Both the EmerState tag output and LED remain ON. • An input value of OFF (zero or less) or NA allows the sensor-user to toggle or turn OFF an emergency state by pressing the Emergency key. However, the EmerState output cannot be toggled OFF inside the application (this requires an Emergency key press at the sensor). *Note:* The MN 800 and standard controllers with Rev.3 or higher firmware have an alternate tag available for this function, the EmerSwitch tag, see below.	Emergency (Heat) LED  The LED follows the Emergency key condition or EmerState output status. ***Note:*** Only one “emer output” tag, either EmerSwitch *or* EmerIconStat, should be used in the application; leave the other tag unconnected. In general, the EmerIconStat tag applies more to Heat Pump applications using emergency heat. The EmerSwitch tag applies more to “general purpose” applications, where both the sensor-user AND application require full access to starting and stopping an “emergency state”.
S5xx only	EmerSwitch EmerSwitch (Available only with MN 800 and standard controllers with Rev.3 or higher firmware)	Class: Digital - Controls Emergency (Heat) Key usage on an MN-S5xx sensor and affects the EmerState tag status, reflected by the Emergency (Heat) LED on the sensor. • An input transition from OFF-to-ON (value over zero) lights the Emergency Heat LED *and* sets the output of the input sensor tag EmerState to ON (100.0). However, the sensor-user *can* toggle (or turn OFF) this emergency state by pressing the Emergency key. If turned OFF, both the Emergency LED and EmerState tag output are OFF. • An input transition from ON-to-OFF (value of zero or less) turns OFF the Emergency Heat LED *and* sets the output of the input sensor tag EmerState to OFF (0.0). However, the sensor-user *can* toggle (or turn ON) this emergency state by pressing the Emergency key. If turned ON, both the Emergency LED and EmerState tag output are ON. *Note:* A not active (NA) input is evaluated as OFF. For example, an NA-to-ON transition is like an OFF-to-ON.	
S4xx, S4xx-FCS, or S5xx	FanStat FanStat	Class: Analog - Allows the Fan icon to appear on the “default” screen of the MN-S4xx, S4xx-FCS, or S5xx LCD. The Fan icon appears (along with from 1 to 3 wavy lines to indicate fan speed) when the input value is ON (value over zero), as follows: • One wavy line if the input value is > 0 but < 33.5. • Two wavy lines if the input value is > 33.5 but < 66.5. • Three wavy lines if the input value is > 66.5. The Fan icon and fan speed lines are not visible if the input value is OFF (zero or less) or NA.	
S4xx, S4xx-FCS, or S5xx	HeatStat HeatStat	Class: Digital - Allows the Heat icon to appear on the “default” screen of the MN-S4xx, S4xx-FCS, or S5xx LCD. The Heat icon appears when the input value is ON (value over zero). Otherwise, the Heat icon is not visible if the input value is OFF (zero or less) or NA.	

Table–3.5 Output Sensor Tags - S-Link Sensor Object. (Continued)

Applies to MN-Sx	WP Tech Appearance (Default)	Output Class / Description	LCD Screen or Sensor Feature Controlled
S3xx, S4xx, S4xx-FCS or S5xx	Display1 Display1 * Can be renamed & reordered in the Hardware Wizard.	Class: Analog - This tag's input value appears on the "default" or primary screen of an S3xx, S4xx, S4xx-FCS, or S5xx sensor (it is typically connected to the sensor tag "RoomTemp"). The Hardware Wizard allows various LCD screen features associated with the display of this value, including unit icons, e.g.; °F, °C, or %, and also (unless the S3) other icons (e.g., Outdoor Air, None, Unoccupied). The Hardware Wizard also determines if this value is formatted for display in tenths or in whole numbers. The sensor display range is any value from -99 to 999. *Out of range* values display at the LCD screen as follows: • Input value is not active (NA) Display = Abn • Input value is below -99.9 Display = -LO • Input value is above 999.9 Display = HI	 (LCD screen of an S4xx or S5xx sensor shown with all icons lit - not typical.)
S4xx or S5xx	Display2 Display2 * Can be renamed & reordered in the Hardware Wizard.	Class: Analog - This tag's input value appears on the second* screen of an MN-S4xx or S5xx sensor (*access from the default screen with one Up-press on the Up/Down Key). The Hardware Wizard provides the same (but separate) display parameters for this value as for Display1.	Same as for Display1.
S4xx or S5xx	Display3 Display3 * Can be renamed & reordered in the Hardware Wizard.	Class: Analog - This tag's input value appears on the third* screen of an MN-S4xx or S5xx sensor (*access from the default screen with two Up-presses on the Up/Down Key). The Hardware Wizard provides the same (but separate) display parameters for this value as for Display1.	Same as for Display1.
S4xx or S5xx	Display4 Display4 * Can be renamed & reordered in the Hardware Wizard.	Class: Analog - This tag's input value appears on the fourth* screen of an MN-S4xx or S5xx sensor (*access from the default screen with three Up-presses on the Up/Down Key). The Hardware Wizard provides the same (but separate) display parameters for this value as for Display1.	Same as for Display1.
S4xx, S4xx-FCS or S5xx	OnStat OnStat	Class: Digital - Allows the ON icon to appear on the "default" screen of the MN-S4xx or S5xx LCD. The ON icon appears only when the input value is ON (value over zero), otherwise it is not visible (if input value is OFF or NA). *Note:* The ON icon is not available in an MN-S4xx-FCS. Instead, this sensor tag controls the "AUTO" icon in the "default" screen. The AUTO icon appears only when the input is ON (value over zero), otherwise it is not visible (if input value is OFF or NA).	MN-S4xx or S5xx: | MN-S4xx-FCS: **AUTO**

Table–3.5 Output Sensor Tags - S-Link Sensor Object. (Continued)

Applies to MN-Sx	WP Tech Appearance (Default)	Output Class / Description	LCD Screen or Sensor Feature Controlled
S2xx, S3xx, S4xx, S5xx	OvrdTime  (Available only with MN 800 and standard controllers with Rev.3 or higher firmware)	Class: Analog - Determines the length of the timed override initiated from the sensor's Override Key, in minutes. The input value is typically any value from 1 to 16382 and is usually sourced from a constant value or NCI. • A negative value, 0, or not active (NA) acts as zero (no override time). In this case, pressing the Override Key at the MN sensor lights the override LED momentarily for approximately 1 second, but no override occurs. • A value of 16383 causes a "forever" override where the override timer does *not* decrement and the override LED remains ON. However, note the override can still be toggled OFF (canceled) and initiated ON at the sensor. The MN-Sxxx Override Key works as follows (regardless of controller firmware revision): • Pressing < 4 seconds initiates an override. • Pressing and holding > 4 but < 8 seconds cancels an active override. • Pressing and holding > 8 seconds causes the controller to send a Service Pin message.	*Note:* During an active override, only an input change to value of either 0 or 16383 is processed immediately. A zero (0) value cancels the active override and prevents subsequent sensor overrides. Value changes besides 0 or 16383 are used in the *next* override initiated from the sensor. See also the two input sensor tags: PBOccMode, OvrdTimeRemain.
S4xx or S5xx	UnoccStat 	Class: Digital - Allows the Unoccupied icon to appear on the "default" screen of the MN-S4xx, S4xx-FCS, or S5xx LCD. The control logic is reverse, that is, the Unoccupied icon appears whenever the input value is OFF (zero or less), otherwise, it is not visible if the input value is either ON (greater than zero) or not active (NA).	

Applying the Sensor Tags (S-Link Object)

Sensor tags are generated by the Hardware Wizard based on the selected model MN sensor and the related sensor options enabled in option tabs. These model-related sensor options determine what can be accessed and adjusted from the MN sensor (setpoints, HVAC mode or fan action), and what can be seen at the MN sensor (LCD screen).

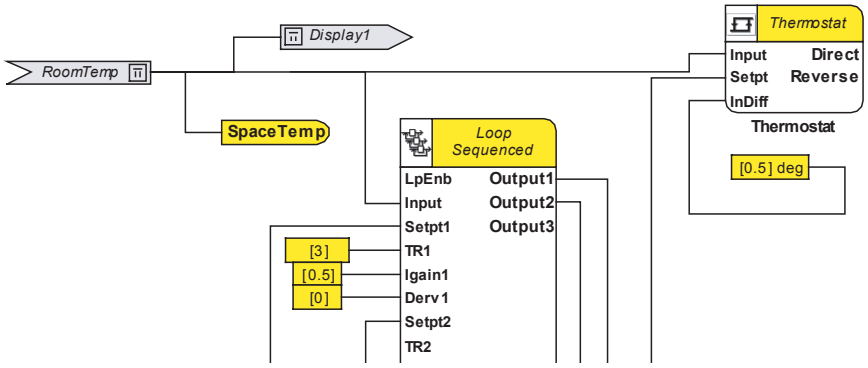
Dialogs within the Hardware Wizard determine initial or default values for many sensor functions that are represented with sensor tags. For example, the initial setpoint associated with an MN sensor's Setpoint (represented by one of the Setpoint tags) is set in the Hardware Wizard, as are allowable adjustment ranges. It is common for the Hardware Wizard to be run several times during the engineering of an application, including the addition or deletion of sensor functions or even changing the MN sensor model. In all cases, the appropriate sensor tags are added to or deleted from the application drawing each time the Hardware Wizard is run. For detailed information on running the Hardware Wizard, refer to the Hardware Wizard chapter in the *WorkPlace Tech Tool 4.0 User's Guide*, F-27255.

The Most Basic Sensor Tags

The input tag RoomTemp is common to all MN sensors and the input tag RelHumidity is common to MN sensors with the relative humidity option. The outputs of these tags are the values measured by the internal sensing elements in the MN sensor. Typically, these outputs are connected to control objects that perform space temperature and space relative humidity control functions.

Note: The MN-S3xx, S4xx, S4xx-FCS, and S5xx models have LCD screens, which often are needed to display room temperature and humidity. For temperature and humidity values to display on the LCD screen, the RoomTemp and RelHumidity outputs must be connected to the appropriate output sensor tag (Display) within the application. The S4 and S5 models can choose from up to four LCD screen configurations, each with a separate Display tag. However, typically only the first display tag (Display1) is connected to RoomTemp. This is because the first Display tag determines the sensor's "default" display, that is, what the sensor displays without any need of keypad input.

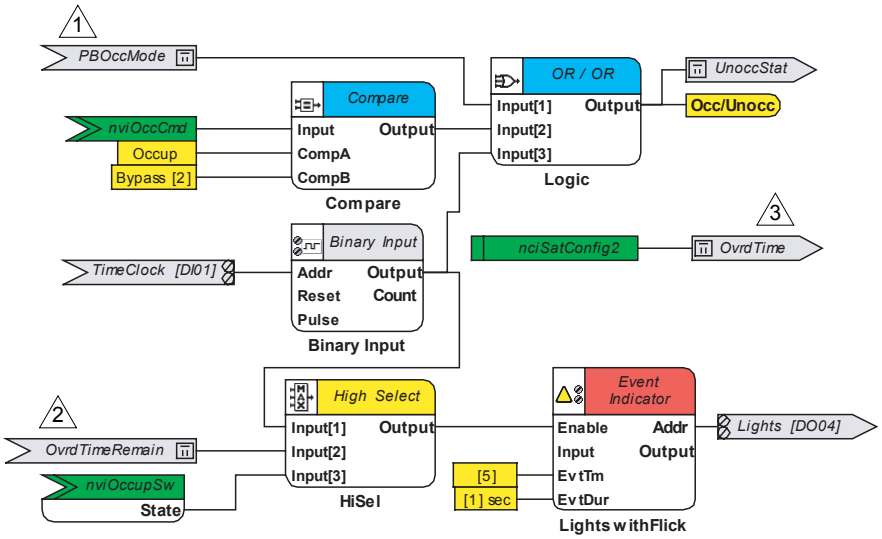
The following example Figure-3.6 shows how the sensor tag RoomTemp has been connected to two control objects, a variable definition tag, and another sensor tag (Display1) in an application that uses an MN sensor model with LCD screen.



Figure–3.6 RoomTemp Sensor Tag Usage, MN-S4 Sensor.

Occupancy Override Sensor Tags

Two input sensor tags relate directly to the Occupancy key (push-button) on the MN-S2xx, S3xx, S4xx, or S5xx sensor. In a Rev.3 or later standard controllers or MN 800 controller, a third output sensor tag is used to establish the time length of the override, in minutes. Figure-3.7 below shows a typical control application.



1 The PBOccMode tag produces an output of digital ON (value of 100.0) during any occupancy-override initiated from the MN sensor. The output returns to OFF (0.0) when the override expires or is canceled.

2 The OvrdTimeRemain tag output is equal to the remaining minutes in a timed occupancy override, from 0 up to 16382 (or from 0 up to 120 if a pre-Rev.3 controller). When an override is initiated from the sensor, the output value goes to the number of assigned minutes, and then counts down each minute until the override either expires (at a value of 0), or it is canceled or re-initiated from the sensor. Note the special case of 16383, below.

3 The OvrdTime tag determines the length of the override in minutes. Typically, this value is sourced from a constant tag or an NCI tag. A value of 16383 defines a "forever" override, meaning the override time does not count down. However, in this mode the override can still be canceled at the sensor, and also restarted again (toggled On and Off) as needed.

Figure–3.7 MN Sensor Override-Related Sensor Tags.

Note: Following a reset to the controller, any previous output value in the two input tags PBOccMode and OvrdTimeRemain is replaced with a value of zero (0) until the sensor Occupancy key is pressed again.

Override Indication/Operation at the MN-S2xx, S3xx, S4xx, or S5xx Sensor

The Override LED is lit (ON) during a sensor-initiated override (Override key is pressed). The Override LED begins flashing when less than 5 minutes remain in the override, and turns OFF when the override expires. If the Override key is pressed during an active override, the override is reset to the full duration. An active override can be canceled by pressing and holding the Override key until the Override LED turns OFF (between four and eight seconds).

If the Override key is pressed and held for more than 8 seconds, the connected controller sends a Service Pin message. This is useful when VAV controllers are being commissioned.

Note: The MN-S4xx-FCS sensors have no “override-to-occupancy” functions.
The Override key and LED have been remapped as the Fan Off/On/Auto key and Fan On/Off LED, respectively. However, a Service Pin message is sent by the attached controller when the Fan Off/On/Auto key is pressed and held for more than 8 seconds.

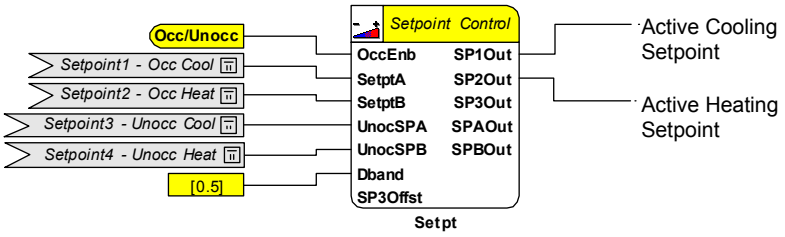
Rev.3 or Later Standard Controller and MN 800 Detail: The override time is set by the value at the output tag OvrdTime (page 65). Typically this is sourced from a constant tag or NCI tag, but might be sourced from an NVI in the application (for a remote cancel). If set to a value of 0, any override is canceled and the Override LED goes OFF.

Setpoint Sensor Tags

A setpoint sensor tag is the result of an enabled Hardware Wizard option for a MN-S3xx, S4xx, S4xx-FCS, or S5xx sensor. The sensor tag outputs a numerical value accessed and adjusted by a MicroNet sensor user. A dialog box within the Hardware Wizard provides an adjustment range (Max and Min) and initial value (Init) for this setpoint. The initial value is active following a download to the controller only until it is adjusted at the MN sensor. Additional Hardware Wizard selections control LCD screen options (numerical formatting, icons) when this setpoint value is accessed at the sensor.

The MN-S3xx and S4xx-FCS have one available setpoint; the MN-S4xx and S5xx each can have up to four setpoints (each created in the Hardware Wizard). Each setpoint results in a separate input sensor tag that has either a default name of “Setpoint” or a custom name that is assignable in the wizard.

The example below Figure-3.8 uses all four setpoint sensor tags.



Figure–3.8 Example Setpoint Sensor Tags (MN-S4 or S5 sensor).

Fan1 and Fan2 Sensor Tags

These tags are results of an enabled Fan1 or Fan2 option in the Hardware Wizard for an MN-S4xx or S5xx sensor (or if an MN-S4xx-FCS, Fan1 option only).

A tag with two outputs is created for each enabled fan option. Each tag has a state output and a value output.

- The state outputs provide a digital value of either 0.0 for AUTO or 100.0 for any other fan action selected from the sensor.
- The value outputs provide a selected analog value corresponding to the sensor-selected fan speed.

Depending on the fan type specified in the Hardware Wizard, the output varies as shown Figure-3.6.

Table–3.6 Fan1.State and Fan2.State Sensor Tag Output Values.

Fan*x* Fan Type	Fan*x* Fan Action	Fan*x*State Output	Fan*x*Value Output
Auto/On/Off[a]	Auto	0	NA
	On	100	NA
	Off	100	0
2-Speed/Off[a]	Low	100	33
	High	100	100
	Off	100	0
2-Speed/Auto/Off[a]	Auto	0	NA
	Low	100	33
	High	100	100
	Off	100	0
3-Speed/Off[a]	Low	100	33
	Med	100	66
	High	100	100
	Off	100	0
3-Speed/Auto/Off[a]	Auto	0	NA
	Low	100	33
	Med	100	66
	High	100	100
	Off	100	0

a.Only equivalent Fan Types without OFF actions are available if controller is pre-Rev.3.

Note: For most fan conditions, output values from Fan*x* sensor tags match the values used in the structured SNVT: SNVT_switch. Refer to page 675 in Appendix B for more details.

HVAC Mode1 and Mode 2 Sensor Tags

These tags are results of an enabled Mode1 or Mode2 option in the Hardware Wizard for an MN-S4xx or S5xx sensor. One tag is created for each.

Each Mode1 or Mode2 sensor tag outputs a numerical value corresponding to a sensor-selected HVAC mode action. Up to four available mode actions can be specified in the Hardware Wizard for selection at the sensor. Selection of a mode produces the following output at the Mode1 or Mode2 sensor tag (Figure-3.7).

Table–3.7 HVAC Mode1 or Mode2 Sensor Tag Output Values.

Mode Action from MN Sensor	Output Value
Auto	0.0
Heat	1.0
Cool	3.0
Off	6.0

Note: Numerical values output from the HVAC Mode sensor tag match some corresponding “mode field” numbers used in the enumerated SNVT, SNVT_hvac_status. Refer to page 671 in Appendix B for more details.

Default Values and Power Cycle Effects

Fan1, Fan2, Mode1, and Mode2 options each have Hardware Wizard selections for the default fan action or mode action. Each default action is active only following an application download to the controller. Any subsequent action (given from the sensor) replaces the corresponding default fan or mode action. The last given fan actions and mode actions are retained following a power cycle to the controller.

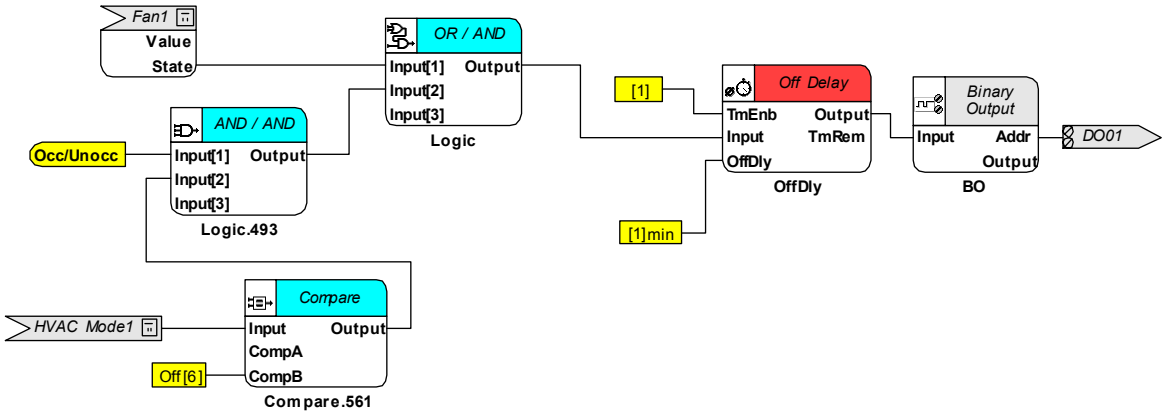
Special Operational Issue for MN-S4 and MN-S5 Sensors (Rev.3 Controllers Only)

Under certain conditions involving Rev.3 controllers, the MN-S4 and MN-S5 sensors do not allow the user to change the Mode1 setting from the sensor. This occurs only when the Mode1 selection is enabled without also enabling either the Mode2 or Fan1 selection. This situation may be resolved through one of the two following methods:

- Enable the Mode2 selection, even if you do not plan to use it, and remove check marks from all the available modes except OFF.
- Enable the Fan1 selection, leaving the rest of the defined values at their defaults, but do not include application programming support for it.

Fan and Mode Tags Example

The following figure shows an example of a fan and mode sensor tag used in a simple application.



Figure–3.9 Example Fan and Mode Sensor Tags (Auto/On Fan Type).

Emergency Heat Sensor Tags

If an MN-S5 sensor is selected in the Hardware Wizard, up to three sensor tags related to the Emergency (Heat) Key and associated LED are created:

- An input sensor tag EmerState (Input Tag).
- An output sensor tag “EmerIconStat.”
- An output sensor tag “EmerSwitch” (MN 800 or standard controller with Rev.3 or later firmware).

EmerState (Input Tag)

Following an application download, the default output of the EmerState tag is Off (0.0). Typically, the output toggles between On (100.0) and Off (0.0) as a result of alternate presses on the MN-S5 sensor’s Emergency key. The sensor’s Emergency LED automatically reflects these states (LED lit if On, LED not lit if Off).

Emer Output Tags

Both of these output tags, EmerIconStat and EmerSwitch, are related to control of the sensor’s Emergency key and LED. If programming a pre-Rev.3 standard controller, only the EmerIconStat tag is available. Only *one* of the two tags, either EmerIconStat or EmerSwitch, should be used in the application—leave the other tag unconnected.

EmerIconStat: The output tag EmerIconStat allows the application to indicate an Emergency state with an On (value over 0.0) to its input.

- An input of On to an EmerIconStat tag lights the Emergency LED *and* sets the output of the EmerState sensor tag to On (100.0).
- If the EmerIconStat input value returns to Off (0.0 or less), the Emergency LED and EmerState tag *remain* On. EmerState can now be toggled Off at the sensor by pressing the Emergency key. Note that EmerIconStat is “level-sensitive”, meaning its input must return Off (0.0) before EmerState can be manually turned Off (at the sensor).
- If the input value to the EmerIconStat tag *remains* On, any press of the Emergency key momentarily toggles the LED Off, but it immediately returns back to On. The EmerState tag output remains On.

EmerSwitch: The output tag EmerSwitch also allows the application to indicate an Emergency state with an On (value over 0.0) to its input. However, it differs from the EmerIconStat tag because the application can also turn Off the EmerState tag (without an Emergency key press). Also, a sensor-user can also turn Off the EmerState tag (and LED), even if the EmerSwitch input remains On. A not active (NA) input is evaluated as OFF.

- An Off-to-On or NA-to-On transition at the EmerSwitch tag lights the Emergency LED *and* sets the output of the EmerState tag to On (100.0).
- An On-to-Off or NA-to-Off transition at the EmerSwitch tag turns Off the Emergency LED *and* sets the output of the EmerState tag to Off (0.0).

Typically, the EmerSwitch tag is recommended for general-purpose applications where full control of the Emergency key is needed by both the sensor-user and from within the application.

Note: (Rev. 4.1 standard controllers and MN 800 only) An active emergency (heat) state is preserved following a power cycle to the controller. An inactive emergency (heat) state is also preserved unless EmerSwitch is ON after the controller restarts.

LCD Screen Sensor Tags

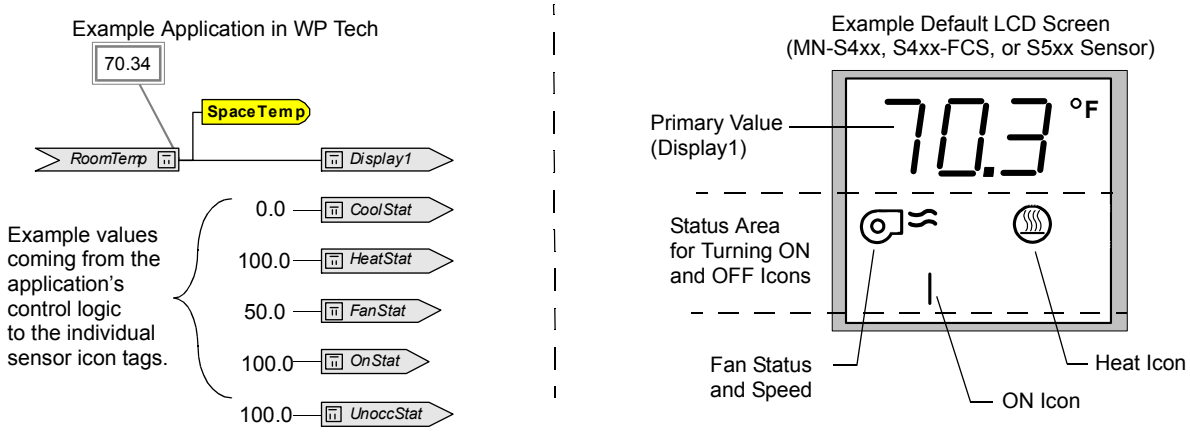
MN sensor models with LCD screens have additional output sensor tags for control of the display (or if an MN-S4xx, S4xx-FCS, or S5xx, individual icons in the *default* display screen). The default display screen appears on the LCD following a 15 second timeout (if S4xx-FCS, 3 second) after no keypad activity.

Individual Icon Tags

The following output sensor tags are automatically created whenever an MN-S4xx, S4xx-FCS, or S5xx sensor is selected in the Hardware Wizard (each tag is available to toggle a specific icon in the sensor's default display screen):

- CoolStat - For the Cool icon.
- FanStat - For the Fan icon.
- HeatStat - For the Heat icon.
- OnStat - For the On (or AUTO) icon.
- UnoccStat - For the Unoccupied icon.

Figure-3.10 shows part of an example application using individual icon tags and the resulting default display screen at the MN-Sx sensor.



Figure–3.10 Individual Icon Tags Control Segments in the Default Display of an S4xx, S4xx-FCS, or S5xx Sensor.

Input value changes to the icon tags dynamically turn Off and On icons in a "status area" below the primary (Display1) value.

Note: Each of the icon tags above *except* UnoccStat requires an ON (value over 0.0) at the input to turn on the associated icon (segment). The UnoccStat requires an input value of OFF (value 0.0 or less) to turn on the unoccupied icon. This provides compatibility with the enumerated SNVT_occupancy, where 0 = occupied and 1 = unoccupied.
Any icon tag left unconnected or with a not active (NA) input has its icon off.

Display (Screen) Tags

Sensor display tags result from enabled Display options in the Hardware Wizard for any MN-S3xx, S4xx, S4xx-FCS or S5xx selected sensor. The S3xx and S4xx-FCS can have a single display (and single sensor display tag); the MN-S4xx and S5xx can have up to four display screens (and sensor display tags), as specified in the Hardware Wizard.

Hardware Wizard dialogs determine how the monitored value appears in the sensor's LCD screen (numerical formatting, additional icons), plus how the associated sensor display tag is named in the application drawing — each display tag may have the default name (Display) or be given a custom name. Hardware Wizard dialogs also allow reordering of monitor values, changing the sequence in which they are accessed at the MN sensor.

Note: If the display or setpoint tags for a MN-S4xx or S5xx sensor are reordered in the Hardware Wizard, the tag's text descriptors (names) remain unchanged. However, WP Tech provides an Edit > Find Object dialog from the menu bar that can be used to find any sensor tag by a default descriptor.

The first Display option (monitor value) listed in the Hardware Wizard is the default display screen at the MN-S4xx or S5xx sensor. Typically, the input to this first display tag is connected to the RoomTemp sensor tag (Figure-3.10), so the MN sensor reads the current room temperature on the default display.

The MN sensor display range for an input value to a sensor display tag is from -99 to 999 as whole numbers or -99.0 to 163.0 with tenths. Out of range values display as follows:

- Unconnected or not active (NA) input value — Display = Abn
- Input value below -99.9 — Display = -LO
- Input value above 999 or 163.0 — Display = HI

MicroNet Sensor Configuration Parameters Not in Sensor Tags

A few MicroNet sensor configuration parameters are *not* graphically represented in a WP Tech application drawing as sensor tags. They must be accessed and modified in WP Tech by running the Hardware Wizard in the opened application drawing. These configuration parameters are:

- The numerical display units (°F or °C) used by the sensor/application
- The offset values applied to the room temperature and relative humidity sensors (default is 0)
- The initial value and allowable adjustment range (Min, Max) for any setpoint adjustable from the sensor
- If a pre-Rev.3 firmware controller, the time (in minutes) for an occupancy override initiated from the MN-S2xx, S3xx, S4xx, or S5xx sensor's Occupancy key (default is 60 and maximum is 120). However, note that MN 800 and Rev.3 and later standard controllers with an MN-S2xx, S3xx, S4xx, or S5xx sensor have the additional sensor output tag "OvrdTime" for this purpose, with an extended range in minutes (up to 16382 for a "timed" override, or a "forever" override by entering 16383).

Other Resource Tags (Standard Controllers)

Profile tags

MicroNet standard controllers (MN 50, 100, 110, 130, 150, 200, and VAV) each contain a particular HVAC LONMARK profile, according to model number. The various network variables (NVs) or Standard Network Variable Types (SNVT) in the controller's profile are represented in the WP Tech application's resource tags as *profile tags*, of three types:

- NCI tags (found initially to the left or bottom of the drawing page)
- NVI tags (found initially to the left of the drawing page)
- NVO tags (found initially to the right of the drawing page)

When engineering a standalone controller application, use of profile tags is optional. These resource tags represent "external" programming boundaries. See Chapter 6, "LONWORKS Network Data Exchange (page 533)", for complete details on profile tags in MicroNet standard controllers and also the equivalent SNVT objects in MN 800 controllers.

Schedule function

MicroNet standard controller models with Rev.3 or later firmware and MN 800 controllers have a resident controller schedule and software clock. The schedule is configured using a Schedule Setup dialog in WP Tech. Both the schedule and the clock are represented in an application drawing by "schedule tags", described in the next section.

Schedule Tags

(Rev.3 or Later Firmware Required)

Usage: Each MN 800 or MicroNet standard controller with Rev.3 or higher firmware has a built-in, 7-day, 4-event-per-day controller schedule. The MN 800 has a capacitor-backed RTC, while MicroNet standard controllers have a software clock. The MN 800 controller clock features adjustable Daylight Savings Time beginning and ending change-over times. Clock and schedule setup in the controller is done in WP Tech using a “Schedule Setup” dialog. An enabled schedule produces this set of “schedule tags” for use in the control logic.

Note: With any MN standard controller, schedule tags require receipt of the valid time on nviTimeStamp to synchronize the controllers software clock. This means the controller’s nviTimeStamp must be bound to a “master timekeeper” device on the LON, typically nvoTimeStamp on an MN 800.

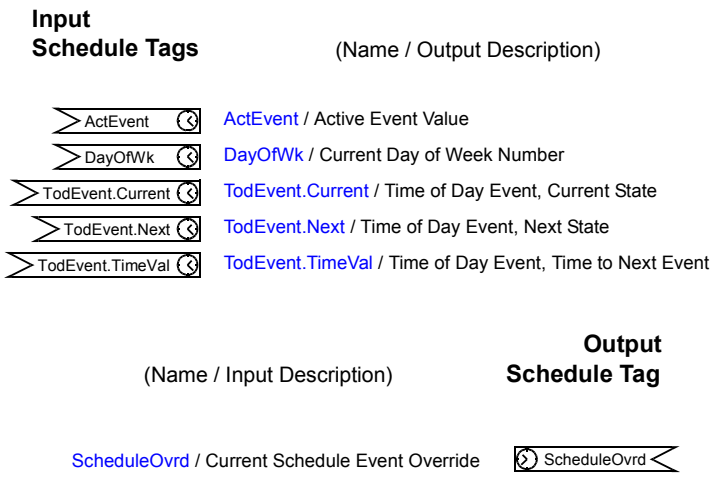
All but one of the schedule tags are input tags, meaning each tag has an *output* that produces a specific schedule value. The sole output tag, ScheduleOvrd, has an input to allow the application to override the current controller schedule event.

The three controller schedule “event” tags can be setup to directly parallel the three NVI event tags, representing this structured NVI:

nviOccSchedule (SNVT _tod_event).

Note: Schedule event tags and event tags for nviOccSchedule operate independently.

WP Tech Representation
(All 6 Schedule Tags)



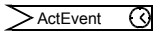
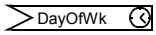
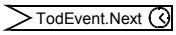
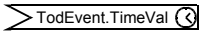
Note: The nviTimeStamp and nvoTimeStamp tags that were in pre-Rev.3 MicroNet standard controller applications are no longer available for use within the control logic. These variables are now specifically used to provide time functionality in the controller.

A value received at the profile’s nviTimeStamp causes the controller’s internal clock to be synchronized to the time received. If the profile’s nvoTimeStamp is bound to another device, that device will receive time and date from the controller’s software clock.

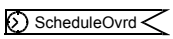
If the controller determines time is not known (clock not running), outputs of all but one of the schedule tags are set to not active (NA). The TodEvent.Current tag output goes to a programmable default value during this “clock not running” period.

Schedule Tags

Table–3.8 Input Schedule Tags - Schedule/Clock Object.

WP Tech Appearance (Default)	Output Class / Description	Default	Valid Values
ActEvent 	Class: Analog - Outputs a value representing the currently active event (1 through 4) in the controller schedule. A value of zero (0) indicates a previous day's event in progress (through midnight). A negative value means the current schedule event is in override mode. For example, if the ActEvent = 2 before override, during an override it is set to -2.	NA	-4 through 4 *Note:* If the previous day's event is overridden, it remains 0.
DayOfWk 	Class: Analog - Outputs a value that represents the current day of the week as a number from 1 to 7 where 1 = Sunday, 2 = Monday, 3 = Tuesday, etc.	NA	1,2,3,4,5, 6, or 7 Remains at not available (NA) if no times are entered in Schedule Setup.
TodEvent.Current 	Class: Analog - Outputs a value that represents the *current* controller schedule event. The format of event values are setup in the Schedule Setup dialog.	As Configured in the Schedule Setup.	As for the selected event data format. See the "Event Data Formats" section on page 78.
TodEvent.Next 	Class: Analog - Outputs a value that represents the *next* controller schedule event. The format of event values are setup in the Schedule Setup dialog.	NA	
TodEvent.TimeVal 	Class: Analog - Outputs the number of minutes until the *next* controller schedule event *change*, up to one week (7 days).	NA	0 to 10079 minutes

Table–3.9 Output Schedule Tag (Schedule/Clock Object).

WP Tech Appearance (Default)	Output Class / Description	Valid Values
ScheduleOvrd 	Class: Analog - Allows the currently active schedule event to be overridden, whenever the input is any valid value. The override remains in effect until either: • The Next Event becomes active. • The input value changes to NA. At controller reset, the input is evaluated as NA until further change.	As compatible with the selected event format. See the "Event Data Formats" section on page 78.

Applying the Schedule Tags

Schedule tags are generated by enabling the schedule option in the Hardware Wizard (when programming an application in an MN 800 or a Rev.3 or later MicroNet standard controller).

Schedule tags reflect the operation of these controller functions:

- Controller Clock (page 77)
- Controller Schedule (Schedule tags) (page 77)

Each function is explained separately in following sections.

Controller Clock

The software clock in each MicroNet standard controller operates using the same format as the structure of SNVT_time_stamp, tracking:

- Year (1900 - 2099)
- Month (1 - 12)
- Day (1 - 31)
- Hour (0 - 23)
- Minute (0 - 59)
- Second (0 - 59)

The controller's schedule task uses the time from the software clock as long as time is valid. The controller clock calculates the day of week based upon the current year, month, and day.

Clock Initialization

The controller clock is synchronized to run whenever one of the following events occurs:

1. An explicit message containing "valid" SNVT_time_stamp information.
2. A "valid" time value is received on **nviTimeStamp** (controller's profile).

Valid Time: The controller considers time "valid" if within these ranges:

Table–3.10 Valid Time Range for Rev.3 MicroNet Controller Software Clock.

Time Field	Valid Values
Year	1900 to 2099
Month	1 to 12
Day	Valid day for Year/Month specified
Hour	0 to 23
Minute	0 to 59
Second	NA (not required)

Note: When synchronized, the controller clock compares the new time value against the current value before making a change. The controller clock is synchronized whenever the new time is "earlier" than the current time by more than 4 seconds, or the new time is "later" than the current time by more than 2 seconds, to prevent undue cycling of time-controlled loads.

Daylight Savings Time Changeover (MN 800)

The Hardware Wizard dialog allows an MN 800 controller's clock to be enabled for automatic time adjustment when beginning or ending Daylight Savings Time. Setup can be either event or occurrence-based. An example of event-based is April 2, providing date setup. Occurrence-based, for example the first Sunday in April, allows for changeover dates to vary from year to year, as needed. When the appropriate "Set Ahead" or "Set Back" time is reached, the controller's clock is automatically advanced or setback one hour.

Note: Do not enable daylight savings time changeover if the controller's clock is synchronized to a network source via the nviTimeStamp SNVT.

Controller Schedule

The controller schedule is a seven-day, four-event-per-day time schedule that is represented in the application by six associated schedule tags.

Five are input tags that produce outputs as follows:

- Current Event (-4 to 4)
- Day of Week (1 to 7)
- Time of Day Event, Current State (data format selectable)
- Time of Day Event, Next State (data format selectable)
- Time of Day Event, Time to Next Event (0 to 10079 minutes)

A single output tag has an input to allow an override of the current schedule event:

- Schedule Override

Note: Event data format can be selected in the Schedule Setup dialog to be compatible with data enumerations in SNVT_occupancy (page 673), if desired.

Schedule Outputs

Controller schedule outputs are set at the top of each minute. If the controller clock is not running (invalid time), the outputs are set as follows:

Table–3.11 Schedule Tag Outputs When Clock is Not Running.

Schedule Tag	Output
TodEvent.Current	Default value assigned in Schedule Setup, or the ScheduleOvrd tag value
TodEvent.Next	NA
TodEvent.TimeVal (to next event)	NA
Day of Week	NA
ActiveEvent (event number)	NA

Event Data Formats

The Schedule Setup dialog in WP Tech allows selection of the type of data format used for the outputs of the current and next schedule tags and for the input of the override schedule tag. Selections include:

Numeric Value	On / Off	Occupied/Unoccupied	SNVT_occupancy
Any value from -163.83 to 16383 or not active (NA), for each possible schedule event.	Off = 0 On = 100	Unoccupied = 0 Occupied = 100	Unoccupied = 1 Occupied = 0 Bypass = 2 Standby = 3 Null = 255 (NA)

Note: Selecting SNVT format after any other format requires reentry of output in each event. This is because, in SNVT_Occupancy mode, 0 = Occupied and 1 = Unoccupied.

TodEvent.Current: If the clock is running, the current day and time are checked for the appropriate programmed event. When an event occurs, the TodEvent.Current tag output is set to the associated value.

TodEvent.Next: The next event is searched in the current day. When found, the TodEvent.Next tag output is set to the associated value. If a next event is not found for the current day, the upcoming day(s) are searched until the next event is found.

TodEvent.TimeVal: Once an event occurs, the remaining time (in minutes) to the next event is calculated and output on the TodEvent.TimeVal tag. This value is continuously updated, providing the time left until the next event.

DayOfWk: The day of week is calculated based on the current time value. Remains at not active (NA) if no times are entered in Schedule Setup.

ActEvent: The active event (number) output is set to 1 from 4 for an event in the current day only. An output of zero (0) indicates an event remains in effect from a previous day. A negative number from -1 to -4 indicates that event has been overridden, for example, -2 means event 2 is currently in override.

Schedule Override

The currently active schedule event is overridden whenever any valid value (anything except not active or NA) is received at the input of the single output schedule tag ScheduleOvrd. The necessary input value depends on the selection of event action format when running the Schedule Setup.

The schedule override remains in effect until the next event becomes active or an NA is received at the input of the ScheduleOvrd tag. If an override is in effect and the controller is reset or power-cycled, the override is canceled until another valid value is received on the ScheduleOvrd tag.

Time Synchronization

Schedule tags require the receipt of a valid time on the nviTimeStamp SNVT to synchronize the internal software clocks of the I/A Series MicroNet controllers. Once the controllers have received a valid time, the controller schedules can be used for standalone operation or for “fall-back” control (control when there is a loss of communication with the device supplying the time clock).

Controller Reset Considerations

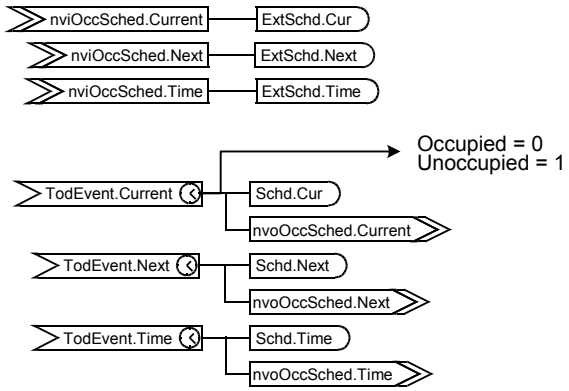
When an I/A Series MicroNet standard controller is reset, downloaded an application, or power-cycled, the current time is lost until the clock is resynchronized. During this time the controller schedule remains inactive, and the outputs of all schedule tags are set to NA except TodEvent.Current, which goes to a default output value. This operation is maintained until the controller receives valid time and date information to re-synchronize the internal clock. When valid time synchronization is received, the software clock calculates the day of week, and the controller schedule is scanned to determine the current event, next event, and time until the next event.

Caution: Due to controller reset considerations, large scheduled equipment (such as chillers and large fans) should have either hardware time delays or software delays (or a combination of both) to protect the equipment. Resets will occur on temporary short term power interruption.

Schedule Tag Example

The output of the TodEvent.Current schedule tag contains the controller's active schedule event (either OFF or ON). This particular application uses both the controller schedule plus an external schedule command received on the LON, via the nviOccSched profile input.

In this example, the event data format for this application has been set in the Schedule Setup to format 3 (SNVT_occupancy). This allows the controller's schedule to be available on the LON, by connecting the three TodEvent schedule tags to the corresponding nvoOccSched profile outputs.



Figure–3.11 Schedule Tags Used in an Application.

Clock Tags (MN 800)

The Schedule Control stencil in MN 800 applications includes a Clock tag. Instances of this tag provide one of the selected components of the Real Time Clock (RTC) (Year, Month, Day, Hour, Minute, or Second). These are input tags that provide current clock information to an application, and behave similarly to the DayOfWk input tag that was placed on the drawing by the Hardware Wizard. For example, at 15 seconds after 11:33 a.m. on May 16, 2001, the values given by these tags would be: Year = 2001, Month = 5, Day = 16, Hour = 11, Minute = 33, and Second = 15.

Only one instance of each of these tags may be placed on the drawing at any time. These tags are typically used when debugging an application.

Controller Memory (RAM and EEPROM)

Each control object requires two types of physical controller memory; non-volatile EEPROM memory and RAM memory. Of the two types, controller RAM usage is most crucial, because it holds a copy of all EEPROM-resident data plus intermediate (real-time) output results.

Note that MicroNet sensor support (MN-Sx) is pre-allocated in a controller's memory requirements. This means selection of any MicroNet sensor model in an application (including none) makes no difference in available RAM.

Standard Controllers

Each MicroNet standard controller has 2K bytes (2048 bytes) of EEPROM and 2K bytes of RAM memory for the controller's "fixed overhead items" plus any programmed control objects. Overhead items include I/O handling routines, MicroNet sensor routines, controller scheduler routines, and LONMARK profile items. Subtracting these items, the remaining controller RAM is available for creation of control objects. The exact amount of available RAM (in bytes) differs only slightly among profile types of MicroNet standard controllers, as shown in Figure-3.12.

Table–3.12 Controller RAM Available for Control Objects, by Profile/Model.

Controller Profile	Standard Controller Models	RAM Available for Control Objects
Fan Coil	MNL-10RF1, -20RF1	1668 bytes
	MNL-5RF2, -10RF2, -11RF2, -13RF2, -15RF2, -20RF2	1676 bytes
	MNL-5RF3, -10RF3, -11RF3, -13RF3, -15RF3, -20RF3	
Heat Pump	MNL-10RH1, -20RH1	1686 bytes
	MNL-5RH2, -10RH2, -15RH2, -20RH2	1694 bytes
	MNL-5RH3, -10RH3, -15RH3, -20RH3	
Roof Top	MNL-10RR1, -20RR1	1678 bytes
	MNL-5RR2, -10RR2, -15RR2, -20RR2	1686 bytes
	MNL-5RR3, -10RR3, -15RR3, -20RR3	
Satellite 1 and 3 (Roof Top variation)	MNL-5RS1, -10RS1, -15RS1, -20RS1	1598 bytes
	MNL-5RS3, -10RS3, -15RS3, -20RS3	
Satellite 2 and 4 (Roof Top variation)	MNL-5RS2, -10RS2, -15RS2, -20RS2	1590 bytes
	MNL-5RS4, -10RS4, -15RS4, -20RS4	
VAV	MNL-V1R1, V2R1, V3R1	1650 bytes
	MNL-V1R2, V2R2, V3R2	1658 bytes
	MNL-V1R3, V2R3, V3R3	

An application in a MicroNet standard controller is limited in total number of control objects only by the controller RAM available and the collective memory requirements of all the control objects. For instance, the theoretical maximum number of control objects in a Fan Coil profile MicroNet controller is 139, based on the unlikely premise that all are math or logic objects, each requiring only 12 bytes of RAM (where 12 bytes x 139 objects = 1668 total bytes). A more likely scenario would be a limit of about 83 total objects, using an average RAM requirement of 20 bytes apiece.

MN 800 Controllers

When programming an MN 800 application, both control objects and SNVT objects consume controller memory. However, because the MN 800 has much more EEPROM and RAM than the standard controllers, memory allocation is not typically an issue.

Drawing Information Storage

In addition to "fixed overhead items" and control objects, WP Tech stores application drawing information in the controllers during download. Application drawing information includes the name of each control object, the location of each object (X, Y coordinates in the application drawing), custom object groupings, and page names. Drawing information is used by WP Tech to generate application drawings from uploaded controller information.

In an MN 800 controller, the drawing information is stored in the controller's EEPROM memory that is not used by overhead items and control objects. Standard controllers have additional EEPROM dedicated to drawing information storage. WP Tech uses a standard controller's dedicated EEPROM plus the controller's EEPROM that is not used by overhead items and control objects for storage of drawing information. Revision two and three standard controllers have 2K bytes of EEPROM dedicated to drawing information. Revision four standard controllers have 6K bytes of dedicated EEPROM.

WP Tech Monitoring

WP Tech provides an application Statistics function that compares the amount of controller RAM required for all the control objects in a control logic drawing against the available RAM in the target controller platform. This function can be quickly accessed from a toolbar icon or as a drop-down menu item at any time in any open application.

The Statistics function displays these RAM-related application statistics:

- Total Memory (in bytes)
- Used Memory
- Available Memory
- Number of Objects

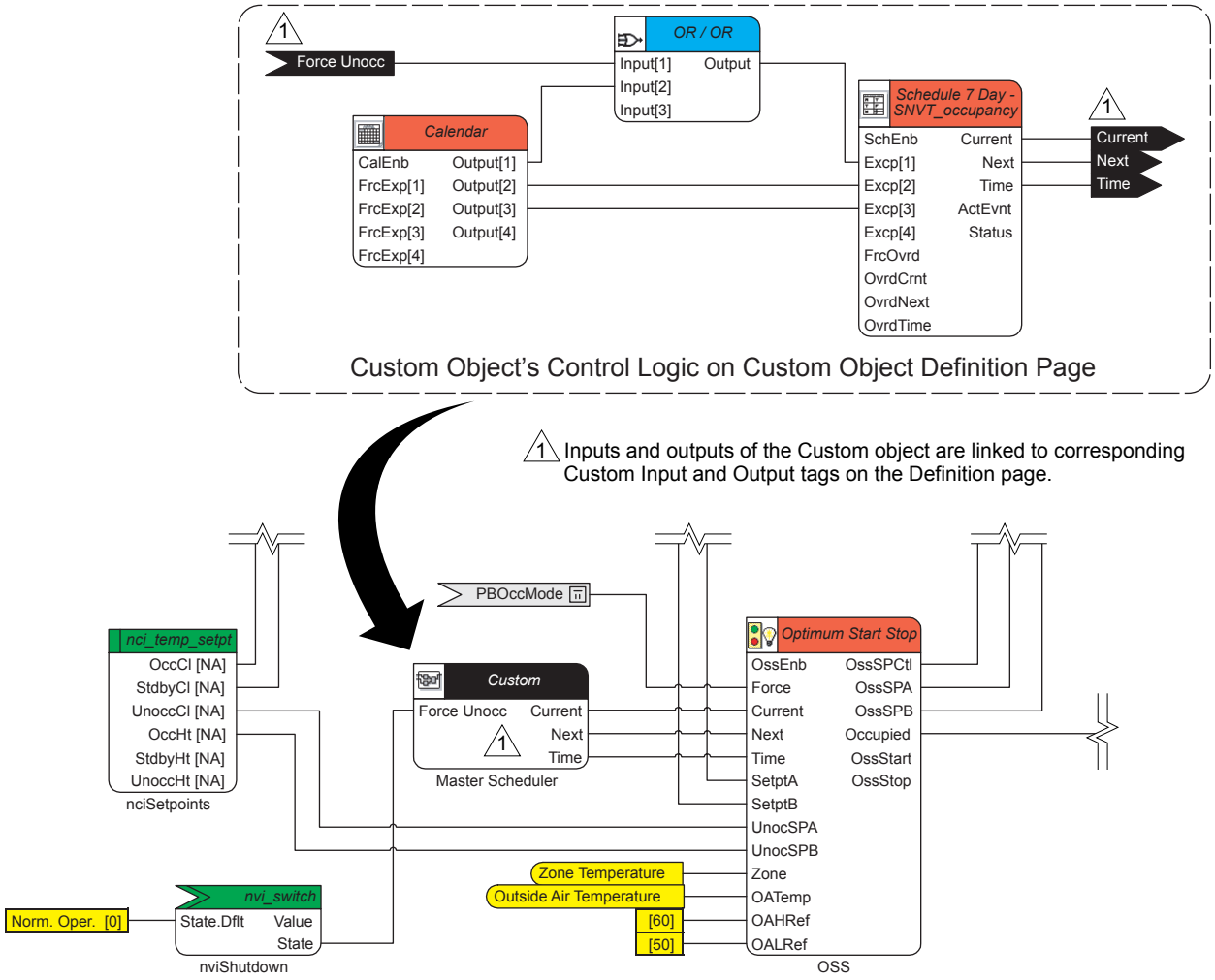
The Statistics function is useful when determining how an application's available memory can best be used, particularly if memory resources are scarce in an application due to a large number of objects.

Information from the Statistics function can be compared to the memory requirements for each type of object, which are listed together in the "Memory Requirements" section in Appendix A of this manual (page 609). Memory requirements for each object are also provided in the individual descriptions for each object in Chapter 5 (page 103).

For more details on running the applications Statistics function, refer to the *I/A Series WorkPlace Tech Tool 4.0 User's Guide*, F-27255.

Custom Objects

The Custom object is a powerful tool for MicroNet controller applications. Each Custom object represents a user-defined, logical grouping of objects defined on a separate, underlying page of the drawing. In other words, Custom objects function as logical "containers" for other objects or subroutines— a way to "simplify" groups of objects into a single reference shape (Figure–3.12). Custom objects are unique because they *are not represented in the controller's database*, but reside only as drawing features in WP Tech applications. The use of custom objects does not decrease the amount of memory used by an application in a controller.



Custom Object on Top Page of Application Drawing

Figure–3.12 Example Custom Object and the Objects it Represents.

There are a number of advantages to using Custom objects. First, a single Custom object can represent an entire control logic sequence, even one containing a large quantity of shapes. This allows you to conserve space on a drawing page, which is especially helpful in more complex applications that use multiple drawing pages. Second, when copying control logic from one drawing to another, it is much easier and more reliable to copy a single Custom object. Third, by using a Custom object that encapsulates a proven, semi-autonomous logic sequence, such as a humidity routine or a grouping of schedules, it helps ensure the overall reliability of the application. Custom objects may be saved on a user stencil for quick availability in other projects and applications.

Logical Boundary

Each Custom object adds an additional page to that application drawing, known as a Custom object Definition page. It is on this page that the Custom object's logic is defined.

Custom Object Rules

Nearly all types of WP Tech shapes may be included on the Definition page of a Custom object, with the following exceptions:

- All "resource tags", such as hardware tags, sensor tags, and schedule tags (i.e. tags generated by the hardware wizard)
- All "Point objects" such as Analog Input, Analog Output, Floating Actuator, etc.
- All "SNVT objects"
- A custom object cannot contain another custom object; nesting of custom objects is not allowed.

These types of objects must be on the *top page* of any control logic drawing, instead.

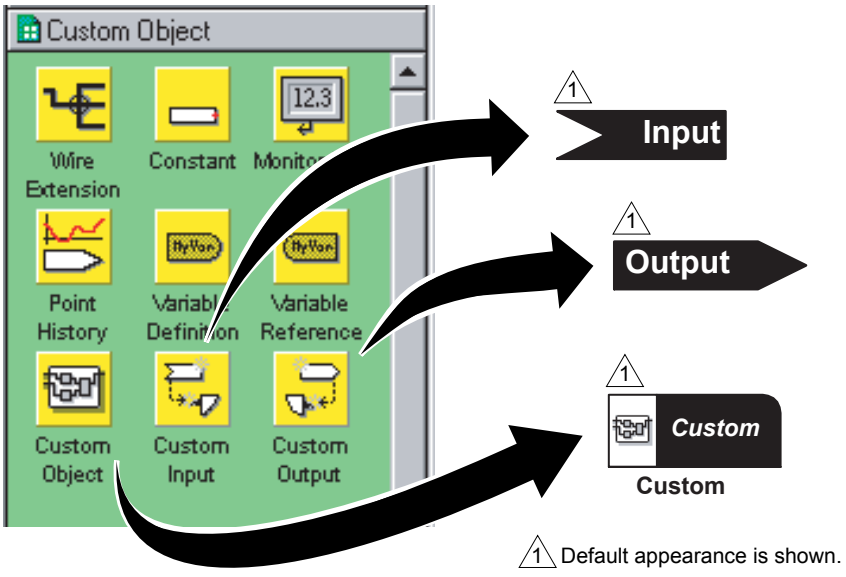
Creating Custom Objects

Custom objects are created in three main steps, as described in the following subsections.

Note: All custom objects should use the same page size as the device definition (top page) of the application drawing.

Defining a Custom Object

The first step in creating a Custom object is to drag a custom object shape onto the drawing from the Custom Object stencil (Figure–3.13). It is then named and defined through the **Customize** function listed in the Custom object's shortcut menu.



Figure–3.13 Custom Object Stencil.

Defining the Custom Object's Inputs and Outputs

Right click the custom object and select **Go to Definition** or click the Custom tab at the bottom of the drawing. On the custom object definition page, inputs and outputs are created by placing and naming Custom Input and Custom Output tags. As these tags are created, their names appear in the Custom object, on the top page of the drawing. The inputs and outputs of the Custom object can be individually shown or hidden, and their order of appearance can be changed, through the **Customize** function.

Engineering the Custom Object's Control Logic

Next, the Custom object's control logic is created on the Custom Object Definition page, in the same way that control logic is constructed on the top page of the drawing. That is, new control logic may be created one shape at a time, or an existing group of shapes may be copied from another drawing page.

Reusing Custom Objects

A powerful aspect of Custom objects is that they may be reused again and again. This may be done by simply copying a Custom object from one open drawing page to another, or by saving them on custom stencils from which they may be copied later. For more information on custom stencils, refer to the "Creating New (Custom) Stencils" section in Chapter 2 of this manual (page 30).

Note: Before reusing a Custom object, be sure its control logic is functional and proven. Any errors that exist in one Custom object would be duplicated many times over when that object is reused in multiple applications, thus complicating the process of correcting that error and increasing the possibility that some of the errors may be missed.

Custom object

Object Usage: Each Custom object that is added creates an underlying Definition page in the active drawing. This underlying page is where the "contained" objects must be dropped and interconnected. The default name for a Custom object (and page) is "Custom." However, by selecting the object and using the right-click **Customize** option, this can be edited to any user-assigned name.

Custom objects have no internal properties. Inputs and outputs are created by placing special Custom Object tags on the Definition page.

The Custom object and the two types of Custom Object tag shapes are found on the Custom Object stencil. The Custom Object tags (with default names) are:



- Custom Input tag



- Custom Output tag

On the Custom Object Definition page, Custom Input tags can be connected to either object inputs or tag inputs. This same behavior applies also to any Custom Output tag, which can be connected to a single object output or tag output.

Also on the Custom Object Definition page, the default name for any Custom object tag can be edited using the right-click **Customize** function. Note the corresponding input or output in the reference shape automatically mirrors the changed name.

Custom objects may be copied within a drawing or saved to a stencil for future use. Using Custom objects promotes a "modular approach" to engineering control logic, which offers many advantages.

WP Tech Shape

Inputs

(none until defined on the Definition Page)



Custom

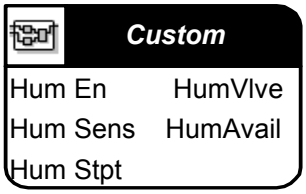
Outputs

(none until defined on the Definition Page)

Custom Object
(default appearance)

Configuration/Status Properties
none

Example Custom Object after Custom Input and Output Tags have been added to Definition Page:



Humidity

Chapter 4
Anatomy of a Control Object

This chapter explains the common characteristics of Invensys control objects. These characteristics include the following:

- Configuration Properties
- One or more Input Properties (inputs)
- One or more Output Properties

This chapter also explains common control object behavior. This behavior includes the following:

- Mixing of Data Types (Analog and Digital)
- Inverted Inputs
- Prioritized Inputs

This material supplements the detailed description provided for each type of control object in Chapter 5, “Control Objects (page 103).”

Object Properties

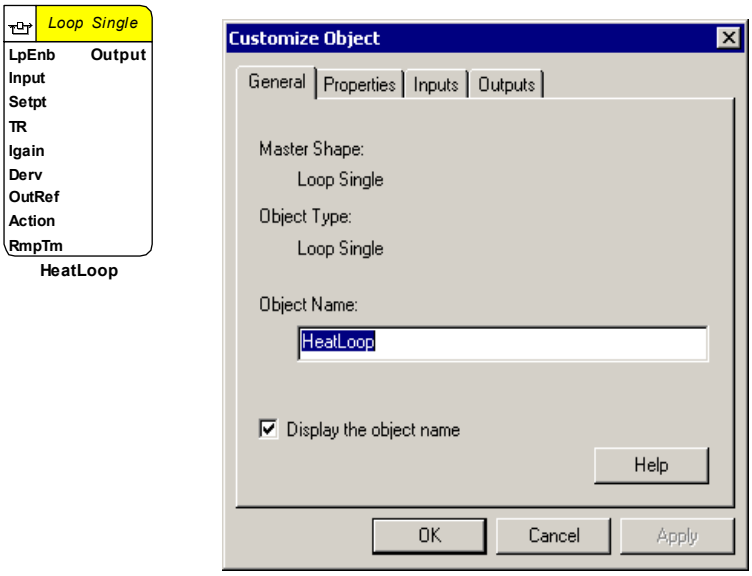
Each control object has three types of properties. The number of properties varies by object type. However, every object type has at least one input (property), one output (property), and two or three configuration properties.

Configuration Properties

Listed below are the three common configuration properties found in every control object:

- Object Name
- Object Description
- Process Time (standard controllers only)

Right-click any control object and select Customize to display the Customize Object dialog box. Figure-4.1. The object name is displayed under the General tab. Click the Properties tab to display the object description and process time properties; many control object types have only these two configuration properties. Several object types have additional configuration properties, which relate directly to the object's control algorithm.



Figure–4.1 Example Object and Configuration Property Editor in WP Tech.

Note: In WP Tech, any number of an object's configuration properties (from all to none) can be set to display below a control object's shape on the control logic drawing. By default, many object types display only the Name property. If desired, other configuration properties can be chosen to display by selecting an object shape, selecting customize from the shortcut menu (right click), and clicking on the Properties tab.

Name and Description

The common properties Object Name and Object Description serve to identify each control object with user-assigned alphanumeric text.

Object Name

This is the unique identifier for the object in the application/controller. When assigning an object name, its uniqueness and its length must both be considered. WP Tech does not allow a duplicate name to be entered.Object names are limited to 31 characters (in the Customize Object dialog box). Invensys recommends that the length be limited to 8 characters or less. This ensures compatibility with future functions in which an object Name may appear in three places:

- On a human-machine-interface (HMI), that is, some future handheld device with an LCD screen that communicates directly (without a PC) to the I/A Series MicroNet controller.
- In object alarm messages routed directly from another type of I/A Series MicroNet controller.
- I/A Series Niagara Binding for SNVTs

Note:

- Each object on an application drawing must have a unique name. WP Tech provides a default Name for every object that is copied to the drawing page. If the same object type already exists on the drawing page, WP Tech appends the new object's default Name with a numerical suffix, to ensure its uniqueness.
- If WP Tech makes a name unique by appending a numerical suffix, the name may be changed by right-clicking on the object and selecting **Customize**. A new or revised Name can then be entered.

Object Description

This is an optional text field that allows a more detailed description of each control object. Unlike Object Name, Description is always a PC-only item, stored within the application drawing file. Objects *retain* very long Descriptions.

Note:

- The Description property will accommodate a large number of characters. However, if this information is to be displayed below the control object, its length should be kept short, so as to maintain the readability of the control logic drawing.
- The Description property is not uploaded or displayed on the drawing of an uploaded application.

Process Time

All WP Tech control objects in a standard controller include the common property Process Time, which affects the frequency of object execution. This property determines how often each control object performs its algorithm relative to other objects in the application.

Note: Process Time is not used in the MN 800. In the MN 800, the controller's enhanced processing power makes the selection of process time unnecessary.

The Process Time setting for each object is not stored in an I/A Series MicroNet controller. Instead, this setting determines how the application compiler in WP Tech organizes objects in the downloadable hex file and how the controller executes these objects.

Note: An I/A Series MicroNet controller's "object engine" executes each object sequentially, one at a time. The WP Tech compiler automatically determines the object execution order for proper sequencing. Each time an object executes, its output(s) update(s).

Process Time

Each control object has three possible Process Times, selected from a drop-down list in the WPT Property Editor. The selections are:

- High
- Medium (the default)
- Low

Process Time is relative to other control objects in the application. Given an application where all control objects have the same Process Time (as when accepting defaults), the controller will execute each object once in each complete object scan. If different Process Times are assigned, objects with a High process time execute twice as frequently than those with a Medium process time, and four times as frequently as those with Low process times. This means the actual frequency of execution for any control object depends on a number of factors, including total number of objects, and the distribution of Process Times among them.

Practical applications for objects to have different Process Times may include scenarios where certain objects are thought to require critical time-based attention (High) or are deemed non-critical in response (Low).

Algorithm-related Configuration Properties

Algorithm-related configuration properties are in addition to the common configuration properties (Name, Description, and Process Time). Many control object types have only the common configuration properties. For example, logic and math type objects have only common properties.

The following object types do have one or more algorithm-related configuration properties:

- Analog Alarm
- Analog Input
- Analog Output
- Analog Output Priority
- Binary Alarm
- Binary Input
- Binary Output
- Calendar
- Curve Fit
- DUI Expander
- Enthalpy
- Fan Speed
- Floating Actuator
- Floating Actuator Priority
- Loop Sequenced
- OSS
- Priority Value Select
- PWM
- PWM Priority
- Ramp
- Schedule 7-Day
- Sensor Input
- Sequence (3)
- Sequence (6)
- Sequence (10)
- Setpoint Control

In most cases, the Property Editor of WP Tech provides a drop-down list for modification of an algorithm-related configuration property. For a few configuration properties, a number must be typed in a field. (The Property Editor displays an acceptable range for these properties.)

Note: If a value typed in a configuration property is outside the acceptable range for that property WP Tech prompts the user for a valid value.

For details on any control object (including all configuration properties), refer to the particular object in Chapter 5. For details on using the Property Editor in WP Tech, refer to the *WorkPlace Tech Tool 4.0 User's Guide*, F-27255.

Input Properties (inputs)

Unlike configuration properties, input properties (inputs) are visible on each control object's shape in a WP Tech application. Most objects have more inputs than outputs. Values received on inputs include variable and constant value data or for input point objects, a physical signal from a controller input (using a physical address assignment). If an object input is unconnected, that input is evaluated as not active (NA).

Physical Address Inputs

A single physical address input is available on four types of control objects, listed in Table–4.1 below. Physical address inputs *should only be connected to* Input Hardware Tags.

Table–4.1 Objects with a Physical Address Input.

Object Type	Compatible Types of Controller Input
Analog Input	UIxx
Binary Input	DIxx, UIxx
DUI Expander	UIxx
Pressure Transducer	Pressure
Sensor Input	UIxx

Controller inputs are represented in WP Tech with Input Hardware Tags. Only one control object can be assigned to any one controller input (only one object can be connected to any one Input Hardware Tag).

Input Data Classes

All other object inputs (beside Physical Address inputs) are data inputs. Each input can be classified as either an analog input or a digital input, depending on the particular control object. This classification serves only to describe:

- The typical format of data received on the input.
- The general method in which the object evaluates that data.

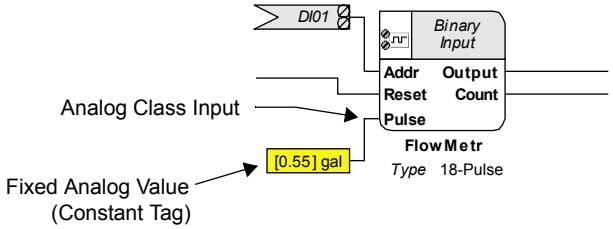
Note: Each class of input (analog or digital) is compatible with any data output, whether that output produces an analog value (-163.83 to 16,383) or a digital value (0.0 or 100.0).

Analog Inputs

If an input to a control object is classified as analog, the object typically receives some analog value at that input, and evaluates that value in some analog fashion.

For example, the Binary Input object has three inputs, one of which is classified as analog (Pulse). This input defines a scaling value used to calculate a delivery rate (output value) at the object's output when the object

is configured for pulse operation. Typically, the Pulse input uses a fixed value, as below Figure-4.2, where a value of 0.55 gallons was entered.

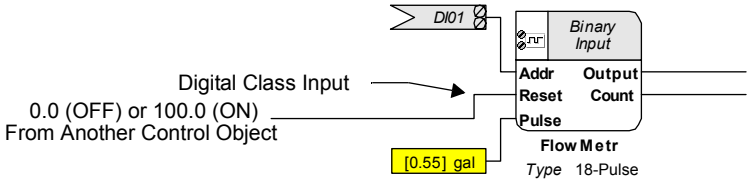


Figure–4.2 An Analog Class Input Typically Receives an Analog Value.

Digital Inputs

If an input to a control object is classified as digital, the input typically receives a digital value (either 0.0 for OFF or 100.0 for ON), and evaluates any received value in a digital fashion. Any value greater than zero is evaluated as ON. Any value less than or equal to zero is evaluated as OFF.

For example, one of the inputs on the same Binary Input object is classified as digital (Reset). This input provides a method to reset both the “Count Output” and (if configured for Pulse) the “output” to zero (0). Typically, this Reset input is connected to a digital class output of another control object, which periodically resets (with a momentary ON) the Binary Input object.



Figure–4.3 A Digital Class Input Typically Receives a Digital Value.

Input Data Sources

Data received on an object input can be one of the following:

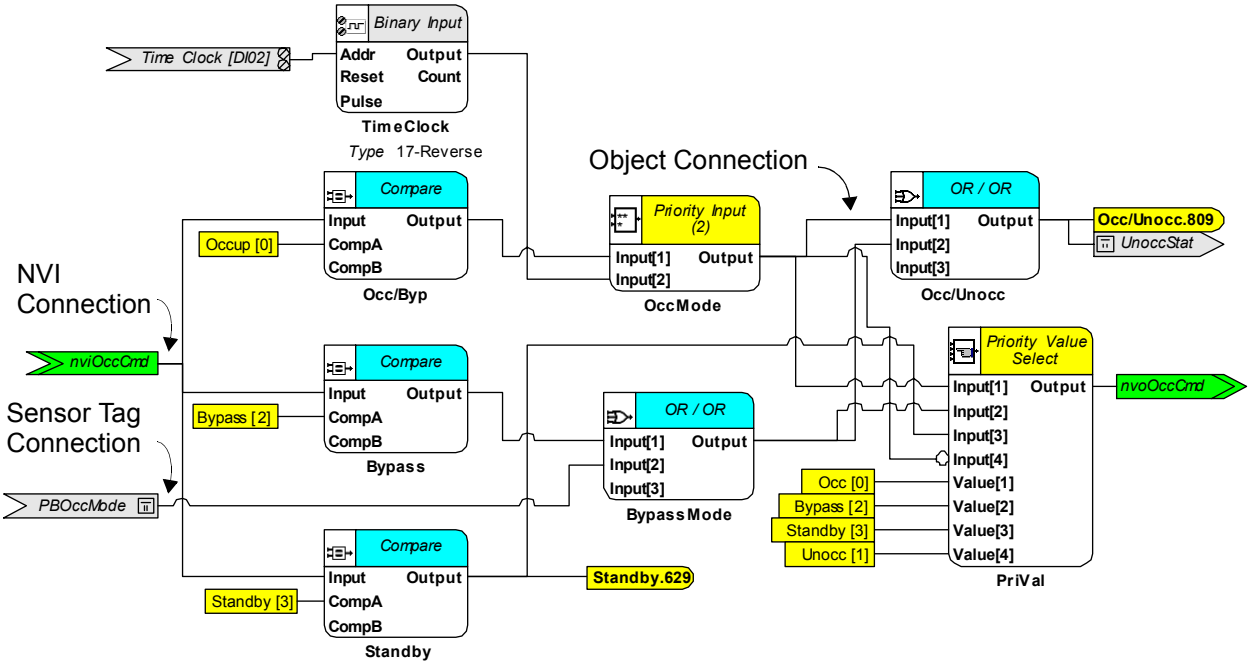
- a variable
- a constant (fixed value)

For any particular object input, there is no set rule as to which type of data source may be used—other than only one source may be used. The graphical shape for each control object in WP Tech ensures this rule through the behavior of the built-in connection wire for each input.

Variable Data Sources

There are three different sources of variable data for an object input:

- An output of another object (object connection).
- An output of a MicroNet resource tag (i.e. sensor tag, schedule tag).
- An output of an input network variable (NVI or NCI).



Figure–4.4 Example Variable Data Sources for Control Object Inputs.

Connection to an object output is the main feature used in most applications. All interobject connections in the application are represented on the control logic drawing in WP Tech.

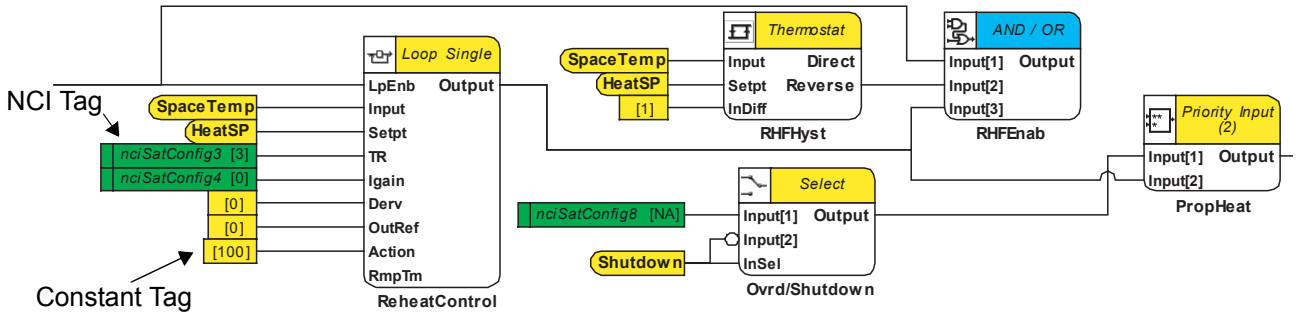
Connection to an input sensor tag allows the selected MicroNet sensor to be used in the application, and defines an aspect of sensor behavior. Again, sensor tag connections are well represented in WP Tech.

Connection of an object input to an NVI allows use of data from another device on the LON, that is, data not generated by the MicroNet controller and its MicroNet sensor. This type of variable input is common in applications that use the controller's network profile. However, it requires a network management tool (program), other than WorkPlace Tech, to make the necessary bindings.

Constant Data Sources

There are two different sources of constant data for an object input:

- A fixed value accessible only in WP Tech (Constant tag).
- A fixed value accessible both in WP Tech and in any LONWORKS network management tool (NCI).



Figure–4.5 Example Constant Data Sources for Control Object Inputs

Connection of an object input to a Constant tag is a technique commonly used in applications. When the application is compiled in WP Tech and downloaded into the MicroNet controller, all of the fixed values in constant tags are copied to RAM and also stored in non-volatile EEPROM.

Values in constant tags can also be individually modified in WP Tech when “Connected” (online with the controller), using a “Write to RAM” function. This diagnostic feature allows quick control logic changes without having to recompile and download the entire application. However, modifications to values in constant tags are not permanent (stored in EEPROM) until the application is recompiled and downloaded.

Note:

- The Write to RAM feature can be used to add a diagnostic capability in an application, beyond the “normal operating” control logic connections. For example, an input to a control object that is typically left unconnected can instead be connected to a Constant tag with a not active (NA) value. This allows a temporary value to be received at the input without a recompile and download. The value may force a disable/enable object function, or fulfill some other diagnostic purpose. Values that are written to RAM are not uploaded with an application. Only values saved to EEPROM are uploaded with an application.
- The value of a constant can be modified by double clicking it or by selecting it and choosing Set Value from the shortcut menu. The Set Value dialog also allows annotation of a constant tag with a prefix and/or suffix (e.g. “Disable [100]” or “[055] gal”).
- Values assigned as constants must be within the range of -163.83 to 16383. The compiler will not detect an error if the value assigned exceeds the limits of the object input to which it is connected.

Connection of an object input to an NVI or NCI is commonly used in applications that use the controller’s network profile. A fixed value in an NVI

or NCI is “network visible” as part of the controller’s profile. This means it can be accessed and modified by a LONWORKS network management tool such as LONMAKER for Windows, I/A Series Niagara, or ICELAN 2000, or a special purpose utility like the MicroNet VAV Flow Balancing Program (in addition to WP Tech).

Note:

- WP Tech provides two global functions for fixed values in an application’s NCI tags or objects: Read NCI Values and Write NCI Values. When read or write NCI values is chosen from the application menu, all NCIs are read or written.
- A single NCI or a group of NCIs can be read or written by selecting them and choosing Read NCI Value or Write NCI Value from the shortcut menu.

Output Properties

Like input properties, output properties (outputs) are visible on each control object’s shape in a WP Tech application. Many functional objects have only a single output, others have two or more.

Outputs represent the data results of each object’s algorithm. Values produced on outputs include analog or digital classes of data, or (for some objects), a physical signal at a controller output (using a physical address assignment).

Physical Address Outputs

One or more physical address outputs are found on several types of control objects. These object types, number of physical address outputs, and compatible types of controller outputs are listed below. Physical address outputs *should only be connected to* Output Hardware Tags.

Table–4.2 Control Objects with Physical Address Output(s).

Object Types	Number of Physical Address Outputs	Compatible Types of Controller Outputs
Analog Output	**1**	AOxx
Analog Output Priority	**1**	AOxx
Binary Output	**1**	DOxx, TOxx
Event Indicator	**1**	DOxx, TOxx
Fan Speed	**3**	DOxx
Floating Actuator	**2**	DOxx, TOxx
Floating Actuator Priority	**2**	DOxx, TOxx
Momentary Start / Stop	**2**	DOxx, TOxx
PWM	**1**	DOxx, TOxx
PWM Priority	**1**	DOxx, TOxx
VAV Actuator	**2**	Actuator, DOxx, TOxx

Controller outputs are represented in WP Tech with output hardware tags. Only one control object can be assigned to any one controller output—this means only one object can be connected to any one output hardware tag.

Note: If desired, any of the output point objects above may be created without assigning output terminal addresses (no connected output hardware tag). The object will function, but only as a “virtual” point without any actual hardware output.

Output Data Classes

Apart from Physical Address outputs, all object outputs are data outputs. An output is classified as either an analog class or digital class output, depending on the particular control object. This classification describes the format of data produced at the output.

- An analog class output produces some varying analog value. Ranges of different outputs vary, but no output value can exceed the MicroNet controller’s numerical range of -163.83 to 16,383.
- A digital class output produces either a 0.0 for OFF, or 100.0 for ON.

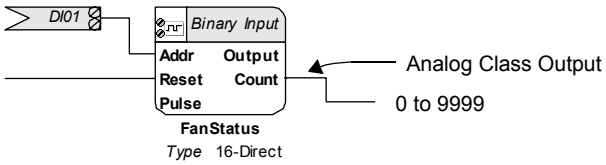
Note: Under various circumstances, control object outputs produce a not active (NA) instead of a valid value (whether analog or digital class). Typically, an NA output is produced because one or more inputs to the control object have an NA, or some abnormal condition has occurred. Refer to each specific control object description for an explanation of when an NA output value is produced.

If the object is an output point object, one or more outputs reflect the physical signal produced at the corresponding controller output point (hardware terminal address).

Analog Class Outputs

If an output is classified as analog, the object output produces an analog value within a range defined by the particular output and object type.

For example, the Binary Input object has two outputs, one of which is classified as analog (Count). This output tracks the number of status changes seen at the hardware input. The specific value range for the Count output is an integer value from 0 to 9,999 (a rollover function resets the count back to zero). An example Binary Input object is shown below Figure-4.6.



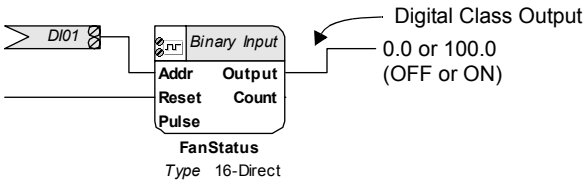
Figure–4.6 An Analog Class Output Produces Some Analog Value.

Other analog class outputs of control objects use the full value range of a MicroNet controller (-163.83 to 16,383), such as with math objects, or some other subset, such as 0.0 to 100.0 as with Loop object outputs.

Digital Class Outputs

If an output of a control object is classified as digital, the output produces only two digital values (either 0.0 for OFF or 100.0 for ON).

For example, one of the outputs on the Binary Input object defaults as digital (Output). When the object is configured as Direct (Type), this output directly reflects the hardware status at the controller input (physical address) — either 0.0 for OFF or 100.0 for ON. An example Binary Input object configured as Direct is shown below Figure-4.7.

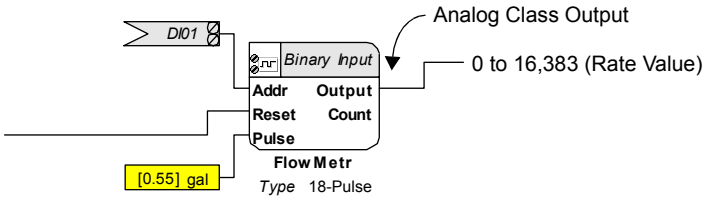


Figure–4.7 A Digital Class Output Produces a Value of Either 0 or 100.

Configurable Class Outputs

A few object types have an output that can be either digital or analog (classified as digital/analog). The Binary Input object is an example. By default, a Binary Input object has a digital Output, as above Figure-4.7.

However, when a Binary Input object is configured as a Pulse (Type), the Output is an analog value based on the value at the Pulse input and the rate of received hardware pulses. An example Binary Input object configured as Pulse is shown below Figure-4.8.



Figure–4.8 A Few Outputs Are Either Digital or Analog (by Object Configuration).

Other objects with digital/analog outputs are the sequence objects (Sequence (3), (6), and (10)), and the Latch object. Each sequence object can be configured for linear, analog, or vernier operation, which defines the function of the outputs. The Latch object can be connected to provide one of these two output functions:

- A digital latch to capture and store a digital (OFF to ON) transition.
- An analog “sample and hold” to capture an analog value.

Common Object Behavior

Each control object type performs a specific routine or algorithm. However, all object types have common behaviors that include:

- Mixing of Data Types.
- Inverted Inputs.

Another common behavior in several object types is:

- Prioritized Inputs.

Mixing of Data Types

Usually when engineering control logic, most connections between control objects are made so that:

- A digital class input is connected to a digital class output.
- An analog class input is connected to an analog class output.

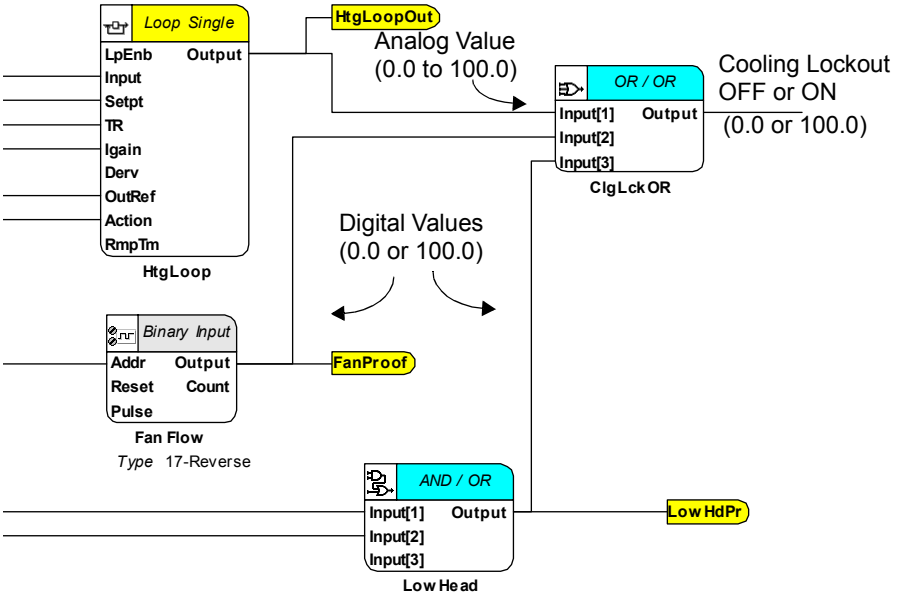
However, because the object data in both classes is numerical, mixed data type connections are also permitted. This means a digital class input can be connected to an analog class output, and the opposite (an analog class input be connected to a digital class output). In some cases, this can provide some utility.

For instance, any logic object evaluates several input conditions and outputs as OFF or ON based on its logic type. All logic objects have three digital class inputs for receiving these conditions.

Consider a scenario where one input condition could be an analog value, such as the output of a Loop Single object used for heating control. The loop object's output will continuously range from 0.00 to 100.00 (%) as the call for heat ranges from no heating to full heating. If this output is connected to an input of a logic object, the value is evaluated digitally as follows:

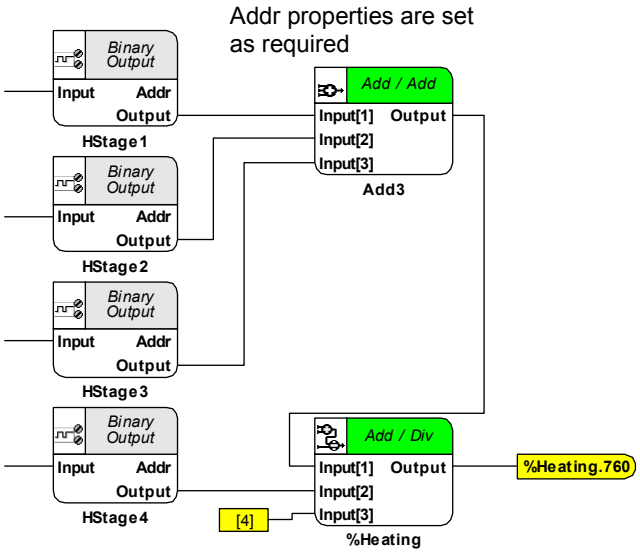
- Value = 0.00, (*No* call for heating), input is OFF.
- Value > 0.00 (*Any* call for heating, from 0.01% to 100.00%), input is ON.

It is possible for this type of digital evaluation to be useful, for example, where a "cooling lockout" function is needed. In this case, the analog value from the heating loop could be "ORed" with digital conditions that would also require disabling of cooling loads, such as states indicating no fan flow or low head pressure, as shown in Figure–4.9.



Figure–4.9 Example of Using an Analog Class Value at a Digital Class Input.

In the same manner, a digital class value of 0.0 or 100.0 may be useful at an analog class input of a control object. An example of this sort of connection is shown below in Figure–4.10.



Figure–4.10 Example of Using a Digital Class Value at an Analog Class Input.

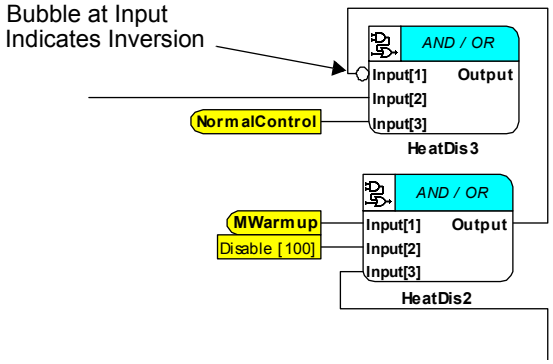
In this example, the digital outputs of each of four Binary Output objects (used for heat stages) are connected to the analog class inputs of two math objects. The math objects are used to calculate an analog value that represents the percentage of active heating.

Inverted Inputs

Any input on a control object can be selectively inverted. If *inverted*, any data at the input is evaluated by the object in a *digital* fashion, whether the input is normally an analog class or a digital class input.

Note: An inverted input does not work with a Constant tag. If an attempt is made to invert the input from a Constant Tag, WP Tech generates an error when the application is compiled. Any other data source is compatible with an inverted input, including an output of an object, a sensor tag, or a network variable (NCI or NVI).

WP Tech allows any number of object inputs to be inverted, and indicates each inverted input with a small bubble Figure-4.11.



Figure–4.11 A Bubble at an Object Input Indicates the Input is Inverted.

By default, none of the inputs of a control object are inverted. However, inputs may be selected/deselected for inversion as needed. In WP Tech, input inversion is found by selecting an object shape, selecting Properties from the shortcut menu, and selecting the inputs tab of the property editor.

Note: If a not active (NA) is at an inverted input, it is passed straight through.

Digital Class Inputs Inverted

Inversion of a *digital* class input is straightforward—the object evaluates that input for a numerical value in an *opposite* manner from normal. This means at an inverted digital class input:

- 0.0 or less is evaluated as ON (100.0).
- greater than 0.0 is evaluated as OFF (0.0).

Digital class inputs such as logic object inputs are commonly inverted. Inversion is particularly useful when opposite state tests are needed at multiple inputs that are each connected to the same object output.

Analog Class Inputs Inverted

Inversion of a *analog* class input also results in a *digital type* evaluation, meaning at an inverted analog class input:

- a value of 0.0 or less is evaluated as ON (100.0).
- a value greater than 0.0 is evaluated as OFF (0.0).

Analog class inputs are less frequently inverted than digital class inputs.

Prioritized Inputs

Several control object types have “prioritized” inputs. These objects are:

- Analog Output Priority.
- Floating Actuator Priority.
- Priority Input (2).
- Priority Input (4).
- Priority Value Select.
- PWM Priority

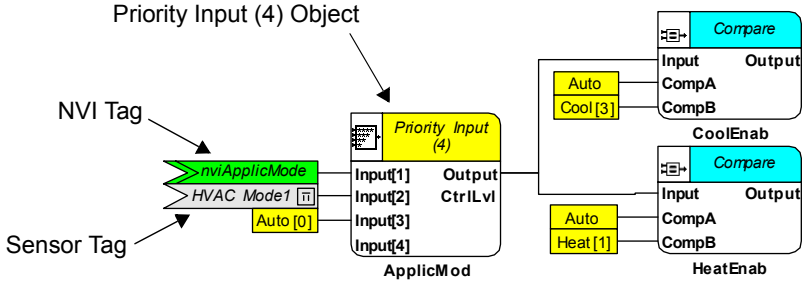
Generally, prioritized inputs mean that a control object has more than one input (typically four) on which to receive data for a single use in the object algorithm. However, only one of these prioritized inputs is used at any time, and it is always the highest priority input that has a valid value.

- In general, a valid value is *any* value *except* a not active (NA).
- Higher priority inputs have lower Input[x] names. This means Input[1] has the highest priority, then Input[2], Input[3], and lastly Input[4].
- Priority inputs appear on object shapes with the highest priority (Input[1]) at the top and lowest priority (Input[4]) at the bottom.

The Priority Input (2) and (4) objects have analog class inputs. Each object simply passes the highest priority (valid) value through to the object’s analog class output. The three prioritized output point objects, Analog Output Priority and Floating Actuator Priority, and PWM Priority are essentially the same as the “non-priority” equivalent objects, but with the added input features of a Priority Input (4) object. The Priority Value Select object uses four priority (digital) inputs to select (and pass to the output) the value present at one of four corresponding analog class inputs.

Priority input objects are often used in applications to provide “fall back” or “contingency” operation in these cases:

- An NVI produces an NA output.
- The value of an NCI is not assigned (NCI has an NA output).
- A MicroNet sensor is disconnected, unavailable, or stops functioning (Sensor tag has an NA output).



Figure–4.12 A Priority Input (4) Object Used in an Application.

Chapter 5
Control Objects

This chapter describes all the various control objects available in I/A Series MicroNet standard and MN 800 controllers. In WP Tech, these objects reside as master shapes on stencils. The chapter begins with two object lists, each showing all the control objects available on the various stencils:

- Objects Grouped by Stencils.
- Objects Grouped Alphabetically.

The remainder of this chapter individually covers each of the 80-plus types of control objects. Object descriptions are in arranged in alphabetical order for easy reference.

For each object, the first page includes a brief description, the WP Tech representation (shape) with all properties listed, and device support and memory requirements. The following pages for each object provide tables for all the object's configuration properties, input properties (inputs), and output properties (outputs). Each object ends with an "Applying the Object" section to discuss the object's behavior and provide examples.

Objects Grouped by Stencils

Control objects are listed here as they are grouped on WP Tech stencils. These stencils contain control objects that are similar in type, for instance logic and math objects are grouped on a “Logic and Math Control” stencil.

IO and Alarm Control

- Analog Alarm
- Analog Input
- Analog Output
- Analog Output Priority
- Binary Alarm
- Binary Input
- Binary Output
- DUI Expander[a,b]
- Fan Speed[b]
- Floating Actuator
- Floating Actuator Priority
- Momentary Start / Stop
- Pressure Transducer[c]
- PWM
- PWM Priority
- Sensor Input[d]
- VAV Actuator[c]

Logic and Math Control

- Abs Sub / Div
- Add / Add
- Add / Div
- AND / AND
- AND / OR
- Average
- Clocked SR
- Compare
- Compare 2[a]
- Count Down
- Count Up
- Curve Fit[a,b]
- Enthalpy[d]
- EXOR
- Filter
- Latch
- MA Volume
- Mul / Add
- Mul / Div
- OR / AND
- OR / OR
- SqRt Mul / Add
- SR Flip-Flop
- Sub / Add
- Sub / Div
- Sub / Mul
- Sub / Sub

Loop and Process Control

- Binary Encoder
- Control Override
- COV Priority[a]
- Demux Select[a]
- High Select
- Interlock
- Limit
- Limit Thermostat[a]
- Loop Sequenced
- Loop Single
- Low Select
- Priority Input (2)
- Priority Input (4)
- Priority Value Select
- Ramp[d]
- Reset
- Select
- Setpoint Control
- Thermostat
- Thermostat 2[a]

Network Variables (MN 800)

- NVI objects[d]
- NVO objects[d]
- NCI objects[d]

Schedule Control (MN 800)

- Calendar[d]
- OSS[d]
- Schedule 7-Day[d]

Timer and Sequence Control

- Dual Delay
- Dual Minimum
- Event Indicator
- Interstage Delay (3)
- Interstage Delay (6)
- Interstage Delay (10)
- Minimum Off
- Minimum On
- Off Delay
- On Delay
- Sequence (3)
- Sequence (6)
- Sequence (10)
- Step Driver[d]

a. Available only in controllers with Rev.3 or later firmware and the MN 800.
b. Not available in VAV controllers.
c. Available in VAV controllers only.
d. Available only in the MN 800.

Objects Grouped Alphabetically

All control object types are listed alphabetically below, along with the I/A Series MicroNet controller support by platform: MN 50, 100, 110, 130, 150, 200, VAV, and 800. A few object types are not supported in all the hardware platforms — these object types are indicated below in **boldface**.

Object Name	MN Controller Support					
	50	100 200	110 130	150	VAV	800
Abs Sub / Div	X	X	X	X	X	X
Add / Add	X	X	X	X	X	X
Add / Div	X	X	X	X	X	X
Analog Alarm	X	X	X	X	X	X
Analog Input	X	X	X	X	X	X
Analog Output		X[a]		X	X[b]	X
Analog Output Priority		X		X	X[b]	X
AND / AND	X	X	X	X	X	X
AND / OR	X	X	X	X	X	X
Average	X	X	X	X	X	X
Binary Alarm	X	X	X	X	X	X
Binary Encoder	X	X	X	X	X	X
Binary Input	X	X	X	X	X	X
Binary Output	X	X	X	X	X	X
Calendar						X
Clocked SR	X	X	X	X	X	X
Compare	X	X	X	X	X	X
Compare 2	X	X[c]	X	X	X[c]	X
Control Override	X	X	X	X	X	X
Count Down	X	X	X	X	X	X
Count Up	X	X	X	X	X	X
COV Priority	X	X[c]	X	X	X[c]	X
Curve Fit	X	X[c]	X	X		X
Demux Select	X	X[c]	X	X	X[c]	X
Dual Delay	X	X	X	X	X	X
Dual Minimum	X	X	X	X	X	X
DUI Expander	X	X[c]	X	X		X
Enthalpy						X
Event Indicator	X	X	X	X	X	X
EXOR	X	X	X	X	X	X
Fan Speed	X	X	X	X		X
Filter	X	X	X	X	X	X
Floating Actuator	X	X	X	X	X	X
Floating Actuator Priority	X	X	X	X	X	X
High Select	X	X	X	X	X	X
Interlock	X	X	X	X	X	X
Interstage Delay (3)	X	X	X	X	X	X
Interstage Delay (6)	X	X	X	X	X	X
Interstage Delay (10)	X	X	X	X	X	X
Latch	X	X	X	X	X	X
Limit	X	X	X	X	X	X

Object Name	MN Controller Support					
	50	100 200	110 130	150	VAV	800
Limit Thermostat	X	X[c]	X	X	X[c]	X
Loop Sequenced	X	X	X	X	X	X
Loop Single	X	X	X	X	X	X
Low Select	X	X	X	X	X	X
MA Volume	X	X	X	X	X	X
Minimum Off	X	X	X	X	X	X
Minimum On	X	X	X	X	X	X
Momentary Start / Stop	X	X	X	X	X	X
Mul / Add	X	X	X	X	X	X
Mul / Div	X	X	X	X	X	X
Off Delay	X	X	X	X	X	X
On Delay	X	X	X	X	X	X
OR / AND	X	X	X	X	X	X
OR / OR	X	X	X	X	X	X
OSS						X
Pressure Transducer					X	
Priority Input (2)	X	X	X	X	X	X
Priority Input (4)	X	X	X	X	X	X
Priority Value Select	X	X	X	X	X	X
PWM[d]	X	X	X	X	X	X
PWM Priority[d]	X	X	X	X	X	X
Ramp						X
Reset	X	X	X	X	X	X
Schedule 7-Day						X
Select	X	X	X	X	X	X
Sensor Input						X
Sequence (3)	X	X	X	X	X	X
Sequence (6)	X	X	X	X	X	X
Sequence (10)	X	X	X	X	X	X
Setpoint Control	X	X	X	X	X	X
SqRt Mul / Add	X	X	X	X	X	X
SR Flip-Flop	X	X	X	X	X	X
Step Driver						X
Sub / Add	X	X	X	X	X	X
Sub / Div	X	X	X	X	X	X
Sub / Mul	X	X	X	X	X	X
Sub / Sub	X	X	X	X	X	X
Thermostat	X	X	X	X	X	X
Thermostat 2	X	X[c]	X	X	X[c]	X
VAV Actuator					X	

a. No analog output on MN 100.
b. Not supported in MNL-V1Rxx model VAV controllers.
c. Controller must have Rev.3 or higher firmware. See ""Controller Firmware Revisions" on page 7.
d. In MNL-11Rxx and MNL-13Rxx controller applications, whenever "Seconds" is selected for the Time Select property, the resolution is 0.1 sec. This accommodates wax motor applications, which require a higher resolution.

Other Objects on Stencils

Annotations Stencil

This stencil contains shapes that may be used to easily add textual information to a drawing. Included are:

- A large selection of standard Visio annotation tools including callouts, text, balloons, stamps, tags, and starbursts.
- Separate WP Tech objects for placing on a drawing, an Invensys background, a controller information block, a sequence of operation, a link to a Microsoft Word document, or one of two variations of the Invensys logo.

Custom Object

This stencil contains three shapes used to create custom objects: a Custom Object, a Custom Input Tag, and a Custom Output Tag. The Custom Object shape on this stencil is used to create a custom object, which represents control logic that is defined by a group of interconnected shapes on an underlying page. Inputs and outputs are assigned to a custom object by applying two other shapes from this stencil, the Custom Input tag and the Custom Output tag. These tags are copied onto the definition page and connected to selected object inputs and outputs on that page. Any number of custom objects may be created and saved, on a user's stencil, for reuse in future applications.

Network Variables Stencil

Objects Not On Stencils

I/A Series MicroNet controllers also contain objects not found on WP Tech stencils. These objects are represented instead by certain types of resource tags. Resource tags are generated by the Hardware Wizard when the specific controller and MN-Sx sensor models for an application are identified. If programming a standard controller with Rev.3 or higher firmware, additional resource tags are available.

Object types represented by resource tags include:

- S-Link Sensor (Sensor Tags) (page 58)
- Schedule/Clock (Schedule Tags) (page 75)

S-Link Sensor Object (Sensor Tags)

Depending on the selected MN-Sx sensor model and options enabled in Hardware Wizard, an S-Link Sensor object is represented by a set of from 1 to 26 sensor tags. Both input sensor tags and output sensor tags exist.

Each sensor tag has a single output or input and a specific behavior related to the selected MicroNet sensor model (MN-S1 through S5). Sensor tags act as "mini-objects", connecting to control objects and other sensor tags to help define the application's control logic and MN sensor behavior. A description of each sensor tag is given in Chapter 3 in the section "S-Link Sensor (Sensor Tags)" (page 58).

Controller Schedule and Clock Objects

Rev.3 or higher MicroNet standard controllers and the MN 800 include objects for the controller's built-in 7-day, 4-event-per-day schedule. The hardware wizard is used to select the schedule time chart object or schedule event table object. The MN 800 has a real-time clock; the other controllers have a software clock (the clock tag object, for the MN 800, is located on the schedule control stencil). In WP Tech, the schedule is represented as a collection of "schedule tags", generated after the schedule is enabled in the Hardware Wizard. Both input schedule tags and output schedule tags exist.

Like sensor tags, schedule tags each have a single output or input for connecting to control objects and other tags in an application. A description of each schedule tag is given in the Chapter 3 section "Schedule Tags" (page 75).

Abs Sub / Div

Object Usage: The Abs Sub / Div object is a three-input math object for use with analog values (AV). The object subtracts Input [2] from Input [1] and divides the absolute value of the result by Input [3].

Output = | AV1 - AV2 | / AV3

At least two valid inputs are required; one at either Input [1] or Input [2], and the other at Input [3].

Device Support: (all firmware revisions)

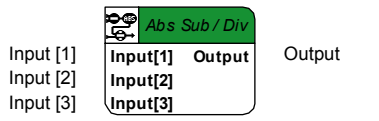
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.1 Abs Sub / Div Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.2 Abs Sub / Div Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input[1]	Input [1]	Class: Analog - The minuend, or the first value in the equation: | **AV1** - AV2 | ÷ AV3	-163.83 to 16383	See the Truth Table for NA input results.
Input[2]	Input [2]	Class: Analog - The subtrahend to the first value (subtracted from the first value in the equation): | AV1 - **AV2** | ÷ AV3	-163.83 to 16383	See the Truth Table for NA input results.
Input[3]	Input [3]	Class: Analog - The divisor. It divides into the previous absolute value in the object's equation: | AV1 - AV2 | ÷ **AV3**	-163.83 to 16383	See the Truth Table for NA input results.

Table–5.3 Abs Sub / Div Object Output Properties

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output indicates the result of the math algorithm. If not active (NA) is present at Input[3], or if NA is present at both Input[1] and Input[2], the output is set to NA.	-163.83 to 16383

Applying the Object

The Abs Sub / Div object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. Its chief distinction is the absolute value function inclusive on the term including the first two inputs.

| AV1 - AV2 | ÷ AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Note: A value of zero at Input[3] causes the output to process a "divide by zero" that sets the output to a maximum (16383) value.

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. Figure-5.4 shows how NA inputs affect the output of the Abs Sub / Div object.

Table–5.4 Abs Sub / Div object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	(| AV1 - AV2 |) ÷ AV3
AV1	AV2	NA	NA
AV1	NA	AV3	(| AV1 |) ÷ AV3
NA	AV2	AV3	(| AV2 |) ÷ AV3
NA	NA	AV3	NA
NA	NA	NA	NA

Add / Add

Object Usage: The Add / Add object is a three-input math object for use with analog values (AV). The object produces an output equal to the sum of all valid inputs.

Output = AV1 + AV2 + AV3

Only one valid input is required to produce a valid output.

Device Support: (all firmware revisions)

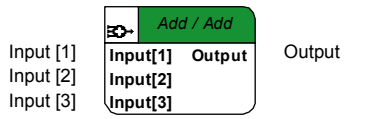
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.5 Add/ Add Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.6 Add / Add Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input[1]	Input [1]	Class: Analog - The first value to summed by the equation: **AV1** + AV2 + AV3	-163.83 to 16383	See the Truth Table for NA input results.
Input[2]	Input [2]	Class: Analog - The second value to be summed by the equation: AV1 + **AV2** + AV3	-163.83 to 16383	See the Truth Table for NA input results.
Input[3]	Input [3]	Class: Analog - The third value to be summed by the equation: AV1 + AV2 + **AV3**	-163.83 to 16383	See the Truth Table for NA input results.

Table–5.7 Add / Add Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output indicates the result of the math algorithm. If not active (NA) is present at all three inputs (Input[1], Input[2], and Input[3]), the output is also set to NA.	-163.83 to 16383

Applying the Object

The Add / Add math object is similar to other three-input math objects in that it processes analog values (AV) and produces an AV output. This object simply adds all inputs together, making their sum the output.

AV1 + AV2 + AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. Figure-5.8 shows how NA inputs affect the output of the Add / Add object.

Table–5.8 Add / Add Object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	Sum (AV1, AV2, AV3)
AV1	AV2	NA	Sum (AV1, AV2)
AV1	NA	NA	AV1
NA	AV2	AV3	Sum (AV2, AV3)
NA	NA	NA	NA

Add / Div

Object Usage: The Add / Div object is a three-input math object for use with analog values (AV). The object divides the sum of Inputs [1] and [2] by Input [3].

Output = (AV1 + AV2) / AV3

At least two valid inputs are required; one at either Input [1] or [2], and the other at Input [3].

Device Support: (all firmware revisions)

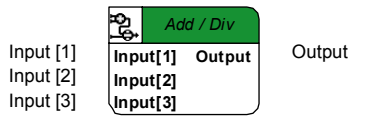
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.9 Add/ Div Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.10 Add / Div Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input[1]	Input [1]	Class: Analog - The the first value summed in the equation: (**AV1** + AV2) ÷ AV3	-163.83 to 16383	See the Truth Table for NA input results.
Input[2]	Input [2]	Class: Analog - The second value summed in the equation: (AV1 + **AV2**) ÷ AV3	-163.83 to 16383	See the Truth Table for NA input results.
Input[3]	Input [3]	Class: Analog - The divisor. It divides into the previous sum term in the object's equation: (AV1 + AV2) ÷ **AV3**	-163.83 to 16383	See the Truth Table for NA input results.

Table–5.11 Add / Div Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output indicates the result of the math algorithm. If not active (NA) is present at Input[3], or if NA is present at both Inputs[1] and [2], the output is set to NA.	-163.83 to 16383

Applying the Object

The Add / Div object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. The equation specific to the Add / Div object is:

(AV1 + AV2) ÷ AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Note: A value of zero at Input[3] causes the output to process a "divide by zero" that sets the output to either a minimum (-163.83) or a maximum (16383) value based upon the results of the first two inputs. A negative result causes the output to be set to the minimum (-163.83) value. A positive result causes the output to be set to the maximum (16383) value.

- Result < 0 sets the output to the minimum (-163.83) value.
- Result > 0 sets the output to the maximum (16383) value.

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. Figure-5.12 shows how NA inputs affect the output of the Add / Div object.

Table–5.12 Add / Div object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	(AV1 + AV2) ÷ AV3
AV1	AV2	NA	NA
AV1	NA	AV3	AV1 ÷ AV3
NA	AV2	AV3	AV2 ÷ AV3
NA	NA	AV3	NA
NA	NA	NA	NA

Analog Alarm

Object Usage: The Analog Alarm object provides for alarm detection of both high and low analog values on the monitored input. Typical monitored values are temperature, pressure, or humidity. Alarm conditions include high alarm, low alarm, return from high alarm, and return from low alarm. The user may specify individual high / low alarm limit values, the deadband for return from alarm, and an alarm delay time. Alarm and return from alarm conditions are indicated at the object outputs and can be stored in the controller's local alarm buffer as alarm message ID numbers, which in turn can be viewed at the controller's MicroNet sensor (MN-S3, S4, S4-FCS, or S5).

Device Support: (all firmware revisions)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4

MNL-11Rxx, -13Rxx
where xx = F2 or F3

MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

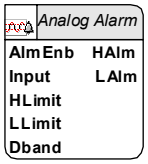
Memory Requirements: (per object)

EEPROM: 22 bytes
RAM: 30 bytes (standard controllers)
8 bytes (MN 800)

WP Tech Representation

Inputs

Alarm Enable
Input
High Limit
Low Limit
Deadband



Outputs

High Alarm
Low Alarm

Configuration Properties

Object Name
Object Description
Process Time
High Alarm Message ID
Low Alarm Message ID
Alarm Delay Time

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.13 Analog Alarm Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.13 Analog Alarm Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
HMsgID	High Alarm Message ID	Class: Analog - A user-defined alarm message ID to be associated with a high alarm condition within the application. A value of 0 (zero) indicates that a message ID is not assigned. A not active (NA) or value outside the defined range causes the High Alarm Message ID to be evaluated as zero.	0	1 to 127	(pre-Rev.3 controllers): A return from high alarm condition adds 128 to the assigned High Alarm Message ID, writing a value between 129 and 255 into the alarm buffer.
LMsgID	Low Alarm Message ID	Class: Analog - A user-defined alarm message ID to be associated with a low alarm condition within the application. A value of 0 (zero) indicates that a message ID is not assigned. A not active (NA) or value outside the defined range causes the Low Alarm Message ID to be evaluated as zero.	0	1 to 127	(pre-Rev.3 controllers): A return from low alarm condition adds 128 to the assigned Low Alarm Message ID, writing a value between 129 and 255 into the alarm buffer.
ADlyTm	Alarm Delay Time	Class: Analog - Defines the length of time (in seconds) that the object must: • Be in an alarm condition before generating an alarm. • Return to a non-alarm condition before generating a return from alarm. An alarm or return from alarm includes generation of network alarm messages and an update of the object alarm output.	0	0.0 to 10,000 seconds	A not active (NA) causes the Alarm Delay Time value to be set to 0.0 seconds.

Table–5.14 Analog Alarm Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
AlmEnb	Alarm Enable	Class: Digital - An input of not active (NA) or ON enables the Analog Alarm function. An input value of OFF causes the algorithm to: • Hold all outputs at their previous state. • Reset the Alarm Delay Time timeouts. • Disable alarm reporting for this object. ***Note:*** If the Analog Alarm object is in an *active alarm state* when an input value of OFF is received, the object *will remain* in the alarm state. Be sure the object *is removed* from the active alarm state before disabling the Analog Alarm function.	On, Off, NA	An input of not active (NA) enables analog alarm function.
Input	Input	Class: Analog - The value at this input is compared against the valid values assigned to the High Alarm Limit and Low Alarm Limit to determine analog alarm conditions. A not active (NA) at this input causes the algorithm to: • Disable alarm reporting for this object. • Hold all outputs at their previous state. • Reset the Alarm Delay Time timeouts.	-163.83 to 16383	

Table–5.14 Analog Alarm Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selections	Notes
HLimit	High Alarm Limit	Class: Analog - Defines the high alarm activation or trip point. This input is continuously monitored and compared to this analog value to initiate the high alarm sequence.	-163.83 to 16383	A not active (NA) causes this input to be ignored, making the high alarm function inactive.
LLimit	Low Alarm Limit	Class: Analog - Defines the low alarm activation or trip point. This input is continuously monitored and compared to this analog value to initiate the low alarm sequence.	-163.83 to 16383	A not active (NA) causes this input to be ignored, making the low alarm function inactive.
Dband	Deadband	Class: Analog - Defines a deadband value which is applied to the High and Low Alarm Limit values to determine the return from alarm trip points. The return from High Alarm trip point value is found by subtracting the deadband value from the High Alarm Limit value. The return from Low Alarm trip point value is found by adding the deadband value to the Low Alarm Limit value.	0.0 to 16383	If unconnected, not active (NA), or a negative value, the Deadband is evaluated as 0.0.

Table–5.15 Analog Alarm Object Output Properties.

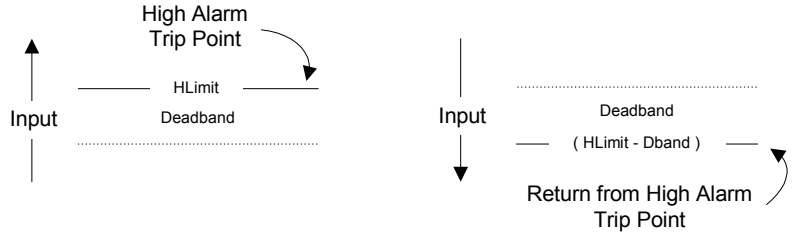
Abbrev.	Name	Class / Description	Valid Values
HAlm	High Alarm	Class: Digital - This output is set to ON whenever the Analog Alarm algorithm has determined a high alarm condition. An OFF indicates that a high alarm condition does not exist.	Normal is OFF (0) Alarm is ON (100)
LAlm	Low Alarm	Class: Digital - This output is set to ON whenever the Analog Alarm algorithm has determined a low alarm condition. An OFF indicates that a low alarm condition does not exist.	Normal is OFF (0) Alarm is ON (100)

Applying the Object

The Analog Alarm object monitors the analog value on its Input and compares it to values at the object inputs for the High Limit and Low Limit. If the monitored value goes outside of either limit, an alarm sequence begins.

High Alarm Sequence

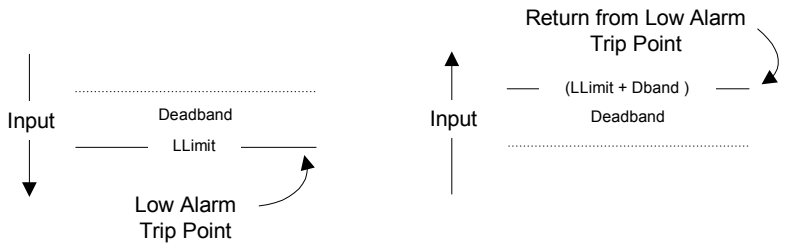
A high alarm sequence is initiated whenever the input exceeds the high alarm trip point (High Limit). A high alarm sequence ends when the input drops below the return from high alarm trip point (High Limit - Deadband).



Figure–5.1 High Alarm Sequence Diagram.

Low Alarm Sequence

A low alarm sequence is initiated whenever the input drops below the low alarm trip point (Low Limit). A low alarm sequence ends when the input raises above the return from low alarm trip point (Low Limit + Deadband).



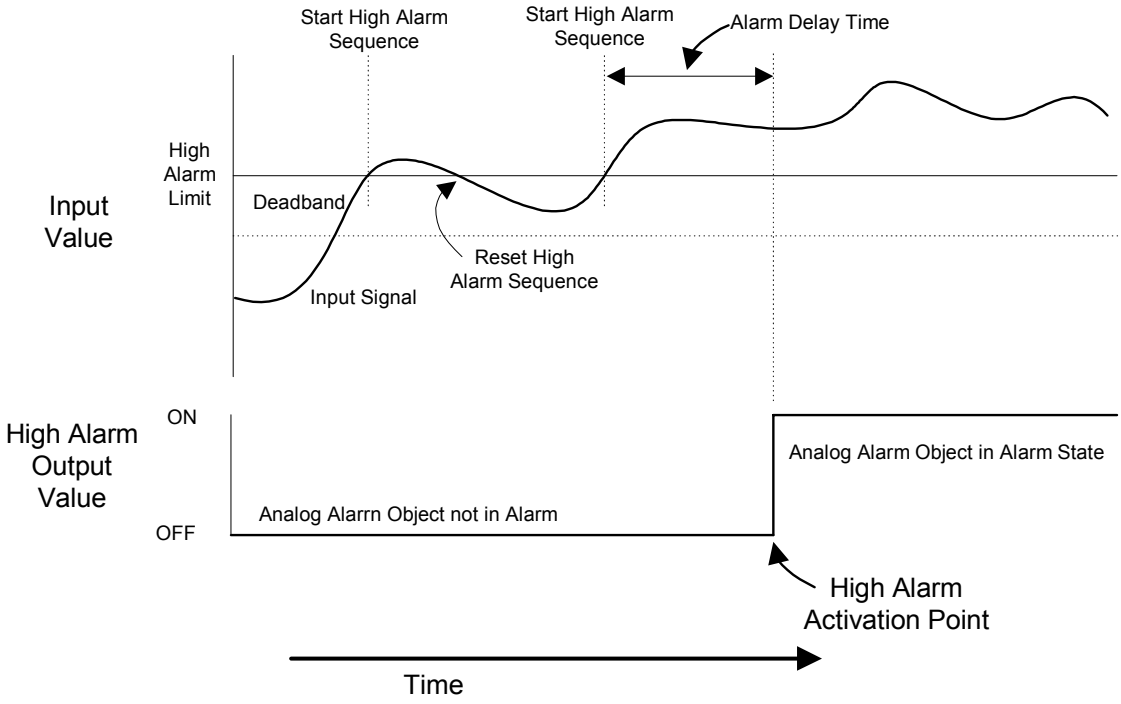
Figure–5.2 Low Alarm Sequence Diagram.

High and Low Alarm Activation

Activation of an alarm occurs whenever an alarm sequence is initiated and the Input remains either above the High Limit (high alarm sequence) or below the Low Limit (low alarm sequence) for a period of time defined in the Alarm Delay Time. An input value which returns below the High Limit (during the high alarm sequence) or above the Low Limit (during a low alarm sequence) prior to expiration of the alarm delay timeout causes the alarm sequence to be reset.

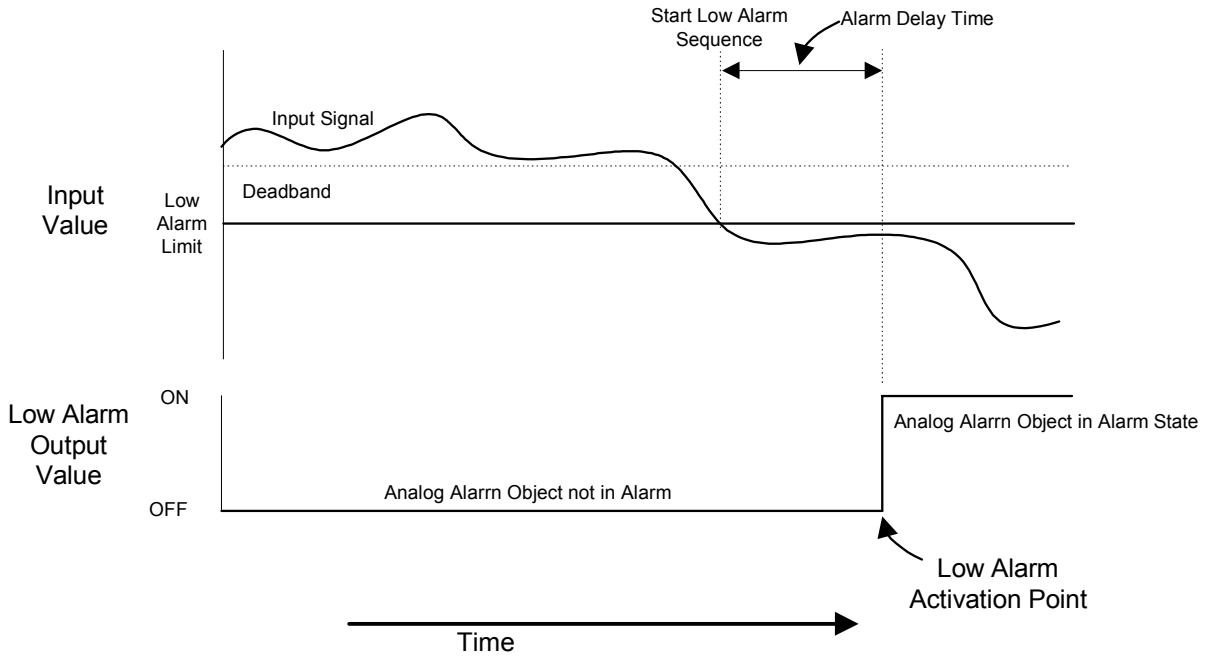
Refer to Figure-5.3 for a graphical example of a high alarm activation and to Figure-5.4 for a graphical example of a low alarm activation.

High Alarm Activation Example



Figure–5.3 High Alarm Activation Example.

Low Alarm Activation Example



Figure–5.4 Low Alarm Activation Example.

Activation of the high or low alarm initiates the following events:

- The high or low alarm is indicated at the object's outputs.
- The high or low alarm may be sent to the controller's alarm buffer.

Analog Alarm Object Outputs

The Analog Alarm object indicates a high alarm condition by setting the High Alarm output to a Digital ON. A High Alarm output of Digital OFF indicates that a high alarm condition does not exist.

The Analog Alarm object indicates the low alarm condition by setting the Low Alarm output to a Digital ON. A Low Alarm output of Digital OFF indicates that a low alarm condition does not exist.

Local Alarm Buffer

Each MicroNet controller has its own local alarm buffer. This local alarm buffer contains the last four reported alarm message ID's within the controller, which can be reviewed at the LCD screen of the MicroNet sensor connected to the controller (MN-S3xx, S4xx, S4xx-FCS, or S5xx models).

The Analog Alarm object reports a High Alarm or Low Alarm activation to the local alarm buffer by sending it the corresponding assigned High or Low Alarm Message ID. The valid range of values for both the High Alarm Message ID and the Low Alarm Message ID is between 1 and 128.

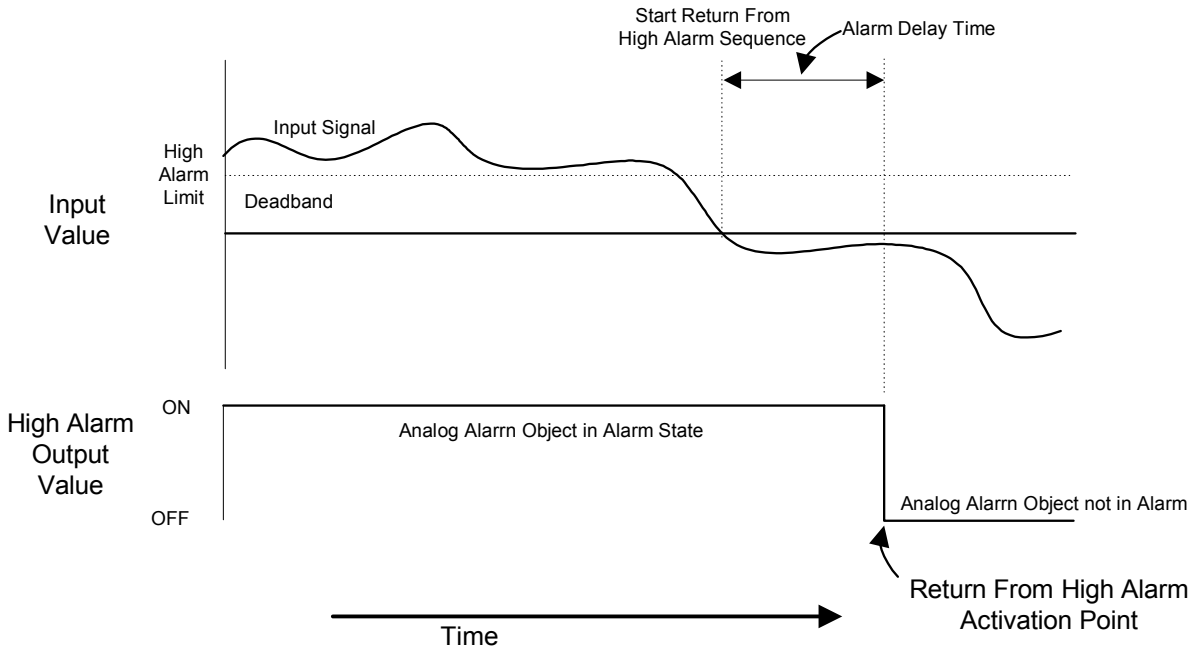
A High or Low Alarm Message ID of zero, not active (NA), or a value outside the defined range indicates that a message ID is not assigned. In this case, Alarm Message IDs are not sent to the local alarm buffer.

Return from High and Low Alarm

A return from alarm sequence occurs after an alarm has been activated and the Input goes either below the High Limit - Deadband (during a high alarm sequence) or above the Low Limit + Deadband (during a low alarm sequence) for a period of time defined in the Alarm Delay Time.
If the input value returns above the High Limit (during a return from high alarm sequence) or below the Low Limit (during a return from low alarm sequence) prior to the expiration of the alarm delay timeout, the return from alarm sequence is reset.

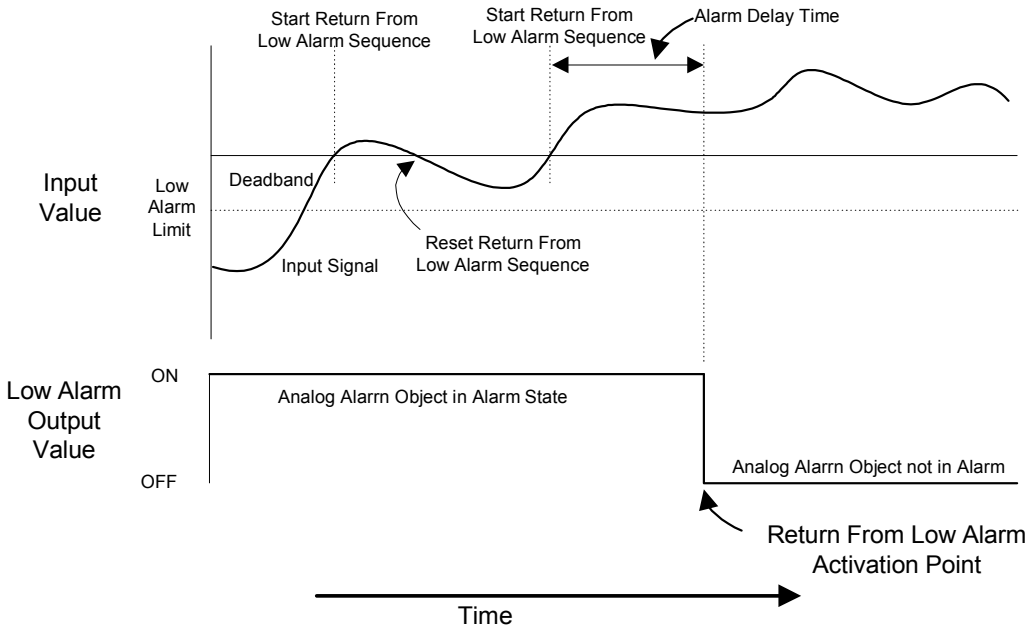
For graphical representations, refer to the examples for a return from high alarm Figure-5.5 and a return from low alarm Figure-5.6.

Return from High Alarm Example



Figure–5.5 Return from High Alarm Example.

Return from Low Alarm Example



Figure–5.6 Return from Low Alarm Example.

A return from a high or low alarm initiates the following events:

- The return from high or low alarm is indicated at the object's outputs.
- The return from high or low alarm may be sent to the controller's alarm buffer.

Analog Alarm Object Outputs

The Analog Alarm object indicates a return from high alarm condition by setting the High Alarm output to a Digital OFF. A High Alarm output of Digital OFF indicates that a high alarm condition no longer exists.

The Analog Alarm object indicates a return from low alarm condition by setting the Low Alarm output to a Digital OFF. A Low Alarm output of Digital OFF indicates that a low alarm condition no longer exists.

Local Alarm Buffer

As described previously, each controller has a local alarm buffer that holds the last four reported alarm message ID's within the controller, which can be reviewed by devices such as MN-S3xx, S4xx, S4xx-FCS, and S5xx sensors.

Note: Rev.3 or higher controllers store only "active" alarms, with Alarm Message IDs in the range of 1 to 127. The next paragraph applies only to pre-Rev.3 controllers.

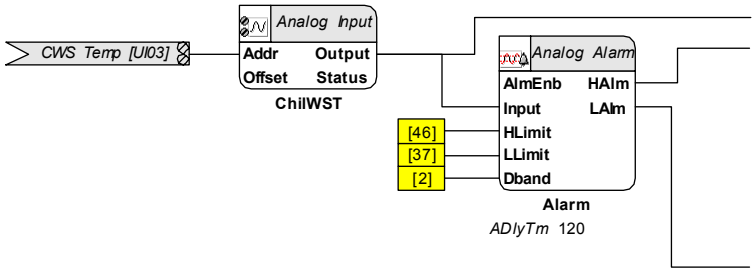
In a pre-Rev.3 controller, when the Analog Alarm object has a return-from-alarm condition, it automatically adds 128 to the corresponding assigned High or Low Alarm Message ID value. This incremented value is then stored in the local alarm buffer as a Return from Alarm Message ID. This makes the valid range of values between 129 and 255 for return from High Alarm Message IDs and return from Low Alarm Message IDs.

A High or Low Alarm Message ID of zero, not active (NA), or a value outside the defined range indicates that a message ID is not assigned. In this case, Alarm Message IDs are not sent to the local alarm buffer.

Example Applications

Constant Alarm Limits

The Analog Alarm object in this example is used to monitor the value of the chilled water supply temperature and report alarms for either a low temperature condition (below 37°F) or a high temperature condition (above 46°F). An alarm deadband of 2°F and an alarm delay time of 120 seconds is used. The Analog Alarm object Input connects to the Output of the Analog Input object used for the chilled water temperature sensor.



Figure–5.7 Analog Alarm Object Example.

In this example, the outputs of the Analog Alarm object are also used in some further digital logic (not shown) in this application. If the Analog Alarm object is in a high alarm, the High Alarm output is ON; otherwise it is OFF. Likewise, if the object is in a low alarm, the Low Alarm output is ON, otherwise it is OFF. A typical result of using these digital outputs may be to cycle Off or On pumps, or close valves.

Alarm Message IDs: Alarm Message ID numbers are assigned by the application programmer. In this example, a High Alarm Message ID of 72 and a Low Alarm Message ID of 22 is assigned to this Analog Alarm object. These non-zero values enable storage of a high or low alarm into the controller's local alarm buffer.

If a high alarm condition occurs, the Alarm Message ID of "072" can be seen from the "ALr" portion of the Diagnostics screens accessible from the controller's MN-S3xx, S4xx, or S5xx model MicroNet sensor. Likewise, if a low alarm condition occurs, the Alarm Message ID of "022" will be visible.

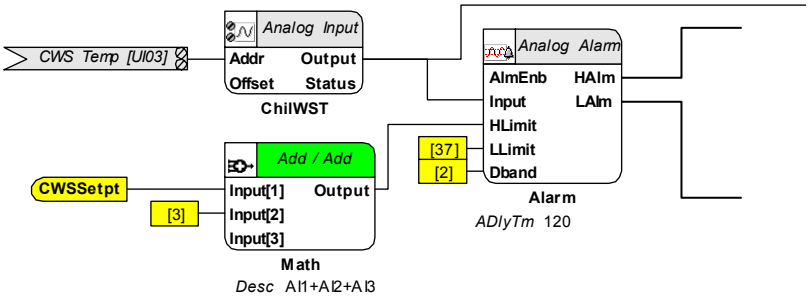
Note: Diagnostics screens of an MN-S3xx, S4xx, S4xx-FCS, or S5xx sensor are brought up by pressing and holding the sensor's entire Up/Down Key for five seconds. Two more Up presses of the Up/Down Key produce the local object alarm buffer, indicated by the flashing "ALr" message followed by two flashes for each of the four possible stored Alarm Message IDs.

Controllers with Rev.3 or later firmware store only message IDs for "active" alarms. Each Alarm Message ID is cleared from the buffer on return from alarm. Return-from-alarm message IDs (those incremented by 128) are stored only in controllers with earlier firmware (MNL-10Rx1, -20Rx1, -VxR1).

Variable Alarm Limit

Because the High Alarm Limit and Low Alarm Limit properties are inputs on the Analog Alarm object, adjustable (vs. constant value) alarm limits can be used. For instance, the previous Analog Alarm object application may be modified to give a high alarm on a deviation from setpoint, rather than a fixed value of 46°F. This allows an earlier warning during high load conditions where the chilled water demand is not satisfied.

In this case, the current chilled water setpoint feeds into an Add / Add object which adds a constant 3°F, and the output of this math object becomes the High Alarm Limit for the Analog Alarm object.



Figure–5.8 Analog Alarm Object Example Using a Variable Alarm Limit.

Note: A deadband setting of 0 will cause alarm state oscillation.

Analog Input

Object Usage: The Analog Input object provides a means for interfacing the application to physical analog type input hardware. This point-type object monitors the assigned hardware input and determines the proper object output value based upon the selected sensor type. Input sensor selections include:

- Thermistor RTD (10k with 11k shunt)
- Balco
- Platinum
- Milliamps
- Volts
- 1 kΩ Resistance (Rev.3 or higher firmware)
- 10 kΩ Resistance (Rev.3 or higher firmware)

Device Support: (all firmware revisions)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

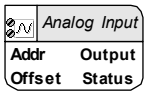
Memory Requirements: (per object)

EEPROM: 20 bytes
RAM: 24 bytes (standard controllers)
4 bytes (MN 800)

WP Tech Representation

Inputs

Physical Address
Offset Calibration



Outputs

Output
Status Flags

Configuration Properties

Object Name
Object Description
Process Time
Input Sensor Type
Input Low Value
Scaled Low Value
Input High Value
Scaled High Value
Filter Constant

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.16 Analog Input Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.16 Analog Input Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Type	Input Sensor Type	Class: Analog - This value defines the sensor type connected to the hardware. Note: Resistance selections (6 and 7) require controllers with Rev.3 or higher firmware (MNL-5Rx2, -10Rx2, -15Rx2, -20Rx2, -VxR2, S1 models).	1	1 - 10k Thermistor 2 - Balco 3 - Platinum 4 - Milliamps 5 - Volts 6 - 1k Resistance 7- 10k Resistance	A 10k Thermistor RTD must have an 11k shunt, such as with any of the TS-87xx-850 series.
LInput	Input Low Value	Class: Analog - Defines the input low value at the actual hardware (Milliamps or Volts) where value at the Output equals the Scaled Low Value. This input is only used for Input Sensor Type selections of Milliamps and Volts. All other selections ignore this property.	0	Milliamps 0.0 to 20.0 or Volts 0.0 to 5.0	A not active (NA) sets the Output to NA and the Status Flags output to ON, indicating an error condition.
LScale	Scaled Low Value	Class: Analog - Defines the output value when the input at the hardware equals the value assigned to Input Low Value. This input is only used for Input Sensor Type selections of Milliamps and Volts. All other selections ignore this property.	0	-163.83 to 16383	A not active (NA) sets the Output to NA and the Status Flags output to ON, indicating an error condition.
HInput	Input High Value	Class: Analog - Defines the input high value at the actual hardware (Milliamps or Volts) where value at the Output equals the Scaled High Value. This input is only used for Input Sensor Type selections of Milliamps and Volts. All other selections ignore this property.	20.0	Milliamps 0.0 to 20.0 or Volts 0.0 to 5.0	A not active (NA) sets the Output to NA and the Status Flags output to ON, indicating an error condition.
HScale	Scaled High Value	Class: Analog - Defines the output value when the input at the hardware equals the value assigned to Input High Value. This input is only used for Input Sensor Type selections of Milliamps and Volts. All other selections ignore this property.	100	-163.83 to 16383	A not active (NA) sets the Output to NA and the Status Flags output to ON, indicating an error condition.
Filter	Filter Constant	Class: Analog -Defines the filter constant or filter factor applied to the input. Maximum Filter is at 0.01 Minimum Filter is at 0.99	1	0.00 to 1.00	An not active (NA), 0.00, or 1.00 bypasses the filtering action.

Table–5.17 Analog Input Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Addr	Physical Address	Class: Analog - Indicates the physical hardware address (input terminal point on the controller) assigned to the Analog Input object.	Dependent on the controller platform selected.	If no physical hardware address is assigned (not active or NA), outputs are also set to not active (NA).
Offset	Offset Calibration	Class: Analog - Defines the value added to or subtracted from the calculated output value prior to placing the value at the actual output.	-163.83 to 16383	If unconnected or with a NA value, the Offset Calibration defaults to 0.0 (no offset applied).

Table–5.18 Analog Input Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The calculated output value for the Analog Input object. This output will indicate not active (NA) whenever the Analog Input object is not assigned a valid physical address or the data from the actual assigned hardware is determined to be not valid. See the operating and diagnostic trip range descriptions defined for each input sensor type.	-163.83 to 16383
Status	Status Flags	Class: Digital - This output is set to ON whenever an error condition is determined by the Analog Input algorithm. An error condition exists whenever the Analog Input object is not assigned a valid physical address or the data from the actual assigned hardware is determined to be not valid. See the operating and diagnostic trip range descriptions defined for each input sensor type.	Normal is OFF (0.0) Error is ON (100.0)

Applying the Object

The Analog Input object monitors the signal received on a universal input (UI) of a LonMark MicroNet controller. The Analog Input object can be assigned to monitor any of the physical UI points on the controller where it resides.

The object's Input Sensor Type assignment configures the hardware terminals to support any of these standard sensor types:

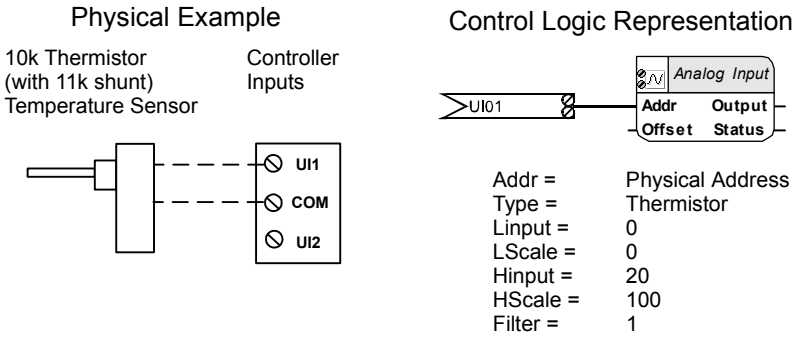
- Resistive Temperature Device (RTD) Sensors
  - Thermistor (10 kΩ with 11 kΩ shunt)
  - Balco
  - Platinum
- 0 to 20 mA
- 0 to 5 V dc
- 1 kΩ Resistance (range from 0 Ω to 1.5 kΩ)
- 10 kΩ Resistance (range from 0 Ω to 10.5 kΩ)

Note: Resistance input selections 1 kΩ and 10 kΩ are available only if programming controllers with Rev.3 or higher firmware.

Thermistor / Balco / Platinum RTD Sensors

Selection of Thermistor (10k), Balco, or Platinum causes the Analog Input object to use controller-resident scaling. This means the input-to-output scaling properties are not used and can be left at default. Offset calibration as well as the filter function may be applied to the output value of the object.

Note: To work properly (without a Curve Fit object), the 10k thermistor RTD must have an 11k shunt resistor, such as with the TS-8500-850 series.



Figure–5.9 Example Analog Input Object for an RTD Temperature Sensor.

Table–5.19 below provides the operating and diagnostic trip ranges for the different RTD sensor types when used with I/A Series MicroNet controllers:

Table–5.19 I/A Series MicroNet Controller Operating and Diagnostic (Not Active) Trip Ranges for RTD Sensor Types.

RTD Type	Operating Range[a]	Valid to NA (Approximations)		Return from NA (Approximations)	
		Low	High	Low	High
Thermistor	-40° to 250°F (-40° to 121°C)	≈ -49°F (-45°C)	≈ 267°F (131°C)	≈ -41°F (-40.5°C)	≈ 251°F (122°C)
Balco					
Platinum	-40° to 240°F (-40° to 121°C)	≈ -44°F (-45°C)	≈ 256°F (124°C)	≈ -41°F (-40.5°C)	≈ 241°F (116°C)

a. General Ranges, refer to the Specification Data Sheet for each particular RTD sensor.

Milliamps / Volts

Selection of Milliamps or Volts causes the Analog Input object to monitor the assigned UI and determine the proper output value based upon the assigned input-to-output scaling. Offset calibration as well as the filter function may be applied to this calculated output value.

For Milliamps, the Analog Input object allows for a 0.0 to 20.0 mA current at the UI terminals, assigned by the Physical Address configuration property. An external 250 ohm shunt resistor must be provided for proper operation.

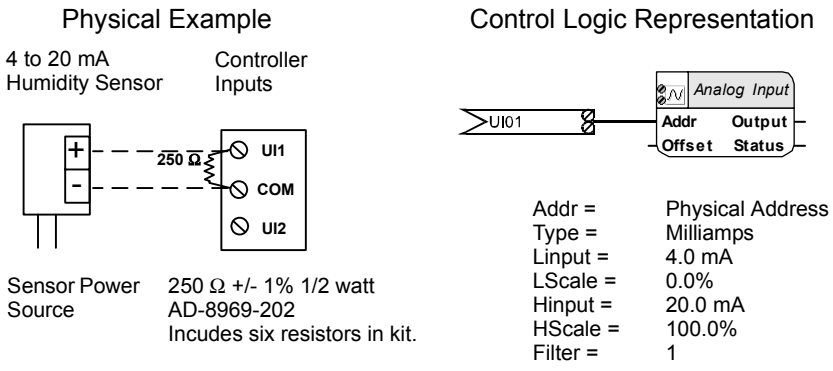
For Volts, the Analog Input object allows for a 0.0 to 5.0 Volt DC signal at the UI terminals, assigned using the Physical Address configuration property. Typical applications include temperature and humidity transmitters, etc. Higher DC voltages, such as 1.0 to 11.0 Volts DC, can be read using a proper voltage divider.

Input-to-output scaling is established by the values assigned to the properties, Input Low Value, Scaled Low Value, Input High Value, and Scaled High Value:

- Setting the Scaled High Value lower than the Scaled Low Value results in inverted or reverse-acting output signals. When doing so, be sure the span between these two values does not exceed 163. If the span is greater than 163, the output will be clipped to a value equal to the Scaled Low Value minus 163.83. For example, if the Scaled Low Value is set to 100 and the Scaled High Value is set to -100, the total span would be 200. Under these conditions, the output would be clipped to 100 minus 163.83, which is -63.83.
- Setting the Input High Value lower than the Input Low Value causes the output to go to not active (NA), and the Status output will indicate an error condition (100.0).

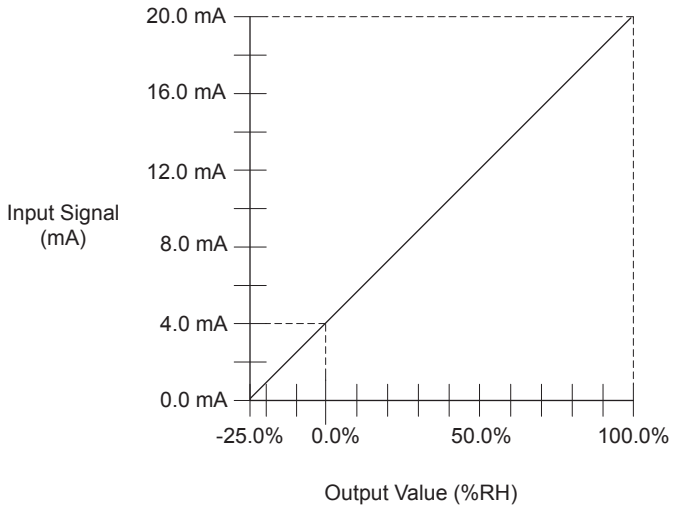
Milliamps

In this example, the property Input Sensor Type has been set to Milliamps. The hardware input (UI) is connected to a humidity transmitter that provides a 4.0 to 20.0 mA signal representing a %RH range of 0.0 to 100.0% RH. The output is scaled to provide 0.0% when the signal at the UI is 4.0 mA and 100.0% when the signal at the UI is 20.0 mA. Note that the shunt resistor and power source for the humidity transmitter must be externally provided.



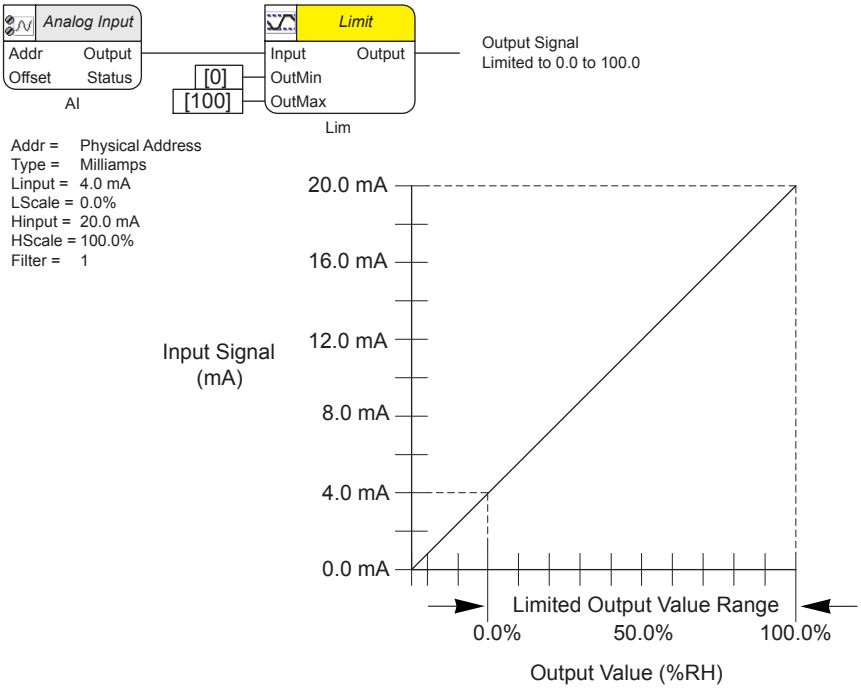
Figure–5.10 Example Analog Input Object Configured for Milliamps.

The relationship of the input signal to the Output value is shown in the following diagram Figure–5.11. The Output value of the object is a percentage of the Input range as established by the Input Low Value / Scaled Low Value and Input High Value / Scaled High Value parameters.



Figure–5.11 Example 4 to 20 mA Input Signal to %RH Output Value Chart.

In applications where the output value needs to be controlled within a defined range, use a Limit Object to limit the active range Figure–5.12.



Figure–5.12 Example 4 to 20 mA Input Signal to a Limited %RH Output Value Chart.

Operating and diagnostic trip ranges (valid value to NA and return from NA) for a universal input (UI) reading current are given in Table–5.20.

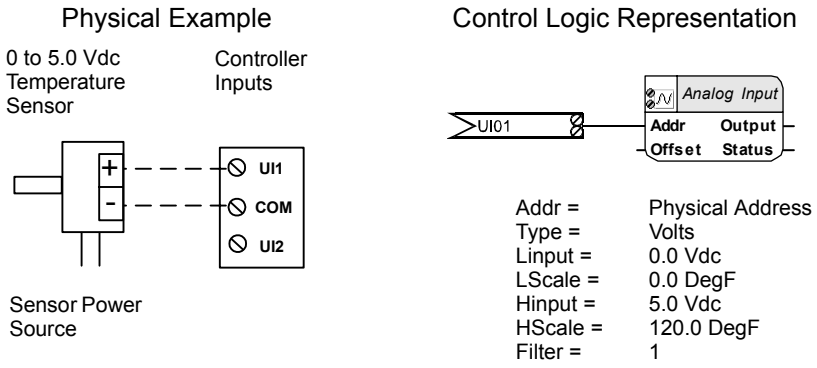
Table–5.20 Current Input Operating Range and Diagnostic Trip Ranges.

Input Range	Valid to NA (approximations)		Return from NA (approximations)	
	Low	High	Low	High
0.0 to 20.0 mA	≈ - 0.1 mA	≈ 20.6 mA	≈ 0.0 mA	≈ 20.3 mA

Note: In the MN 800, the Status Flags output will not indicate an over or under range error condition if “milliamps” is selected for Input Sensor Type. An error condition will still be generated if any of the four configuration scaling constants is NA, or if the Input High Value is set lower than the Input Low Value.

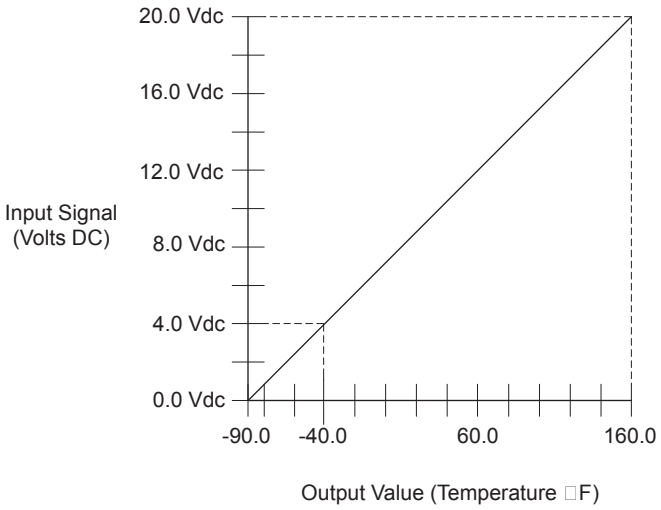
Volts

In this example, the property Input Sensor Type has been set to Volts. The hardware input (UI) is connected to a temperature transmitter that provides a 0.0 to 5.0 Volts DC signal representing a Degrees F temperature range of 0.0 to 120.0°F. The output is scaled to provide 0.0°F when the signal at the UI is 0.0 Vdc and 120.0°F when the signal at the UI is 5.0 Vdc. Note that the power source for the temperature transmitter is provided externally.



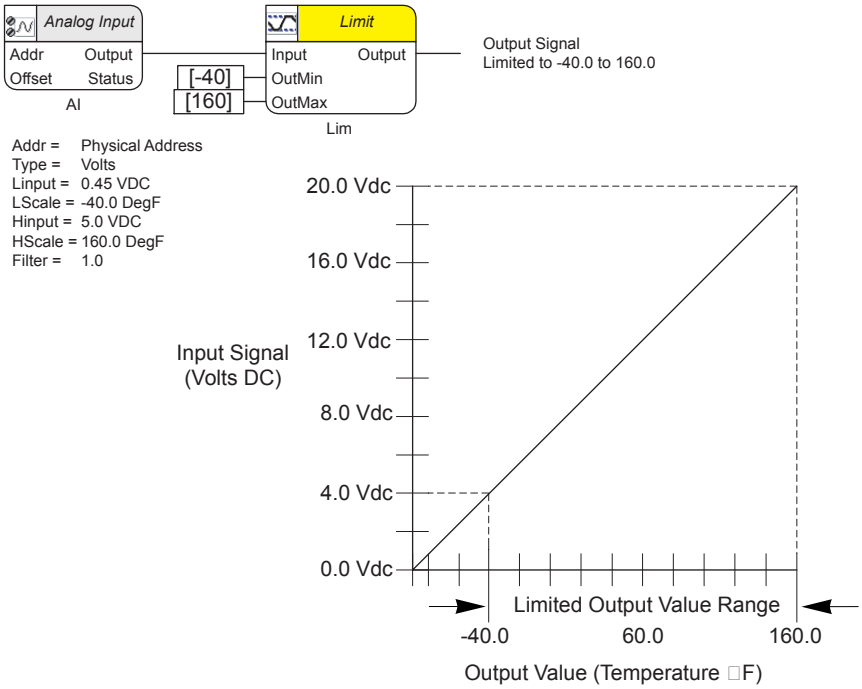
Figure–5.13 Example Analog Input Configured for 0 to 5 Vdc Device.

The relationship of the input signal to the Output value is shown in the following diagram. The Output value of the object is a percentage of the Input range as established by the Input Low Value / Scaled Low Value and Input High Value / Scaled High Value parameters.



Figure–5.14 Example 1 to 11-Volt Input Signal to Deg F Output Value Chart.

In applications where the output value needs to be controlled within a defined range, use a Limit Object to limit the active range Figure–5.15.



Figure–5.15 Example 1 to 11 Volt Input Signal to Limited Deg F Output Value Chart.

The status flags output will not indicate an over or under range error condition if “Volts” is selected for Input Sensor Type. An error condition will occur if any of the four configuration scaling constants are NA.

Operating and diagnostic trip ranges (valid value to NA and return from NA) for a universal input (UI) reading voltage are shown below.

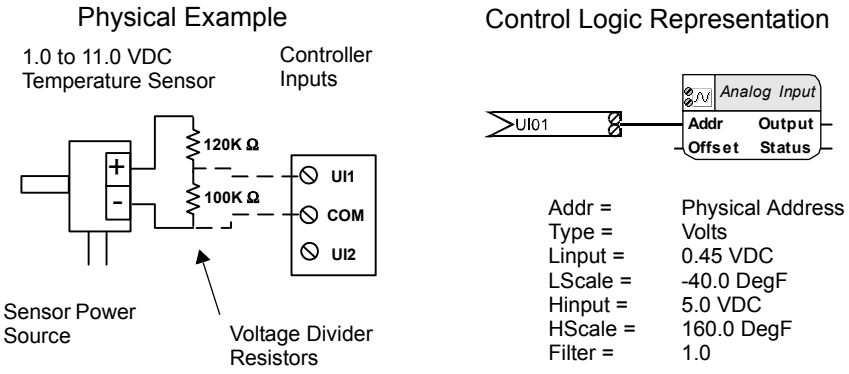
Table–5.21 Voltage Input Operating Range and Diagnostic Trip Ranges.

Input Range	Valid to NA (approximations)		Return from NA (approximations)	
	Low	High	Low	High
0.0 to 5.00 Vdc	≈ - 0.25 Vdc	≈ 5.15 Vdc	≈ 0.0 Vdc	≈ 5.075 Vdc

Note: In the MN 800, the Status Flags output will not indicate an over or under range error condition if “volts” is selected for Input Sensor Type. An error condition will still be generated if any of the four configuration scaling constants are NA, or if the Input High Value is set lower than the Input Low Value.

Sensors or transmitters delivering over 5.0 Vdc can be monitored by using a voltage divider made from two resistors. The resistors must be sized to reduce the transmitter output voltage to within the 0.0 to 5.0 Vdc input range.

Figure–5.16 shows the use of a temperature transmitter which provides a 1.0 to 11.0 Volts DC signal representing a temperature range of -40.0 to 160.0°F. In this example, 120k and 100k resistors provide the necessary voltage divider. The output is scaled to provide -40.0°F when the signal at the UI is 0.45 Vdc and 160.0°F when the signal at the UI is 5.0 Vdc.



Figure–5.16 Example Analog Input Configured for a 1 to 11.0 Vdc Device.

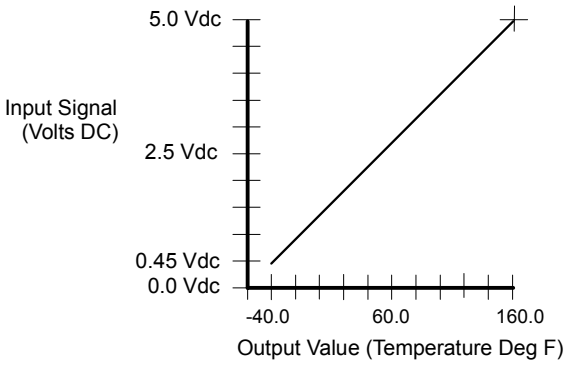
Input Low and Input High Value Calculations: Equations for finding Input Low Value and Input High Value settings are:

Input Low Value = Device Low Voltage Signal x (Voltage Divider Ratio)
Input High Value = Device High Voltage Signal x (Voltage Divider Ratio)

For the settings in the example above, the equations used were:

Input Low Value = 1.0 Vdc [100k ÷ (100k + 120k)] or 0.45 Vdc
Input High Value = 11.0 Vdc [100k ÷ (100k + 120k)] or 5.0 Vdc

The relationship of this example’s hardware Input to the Output value is shown in Figure–5.17 below.



Figure–5.17 Example 1 to 11 V dc Input Signal to Deg F Output Value Chart.

Resistance (1kΩ and 10kΩ)

Selection of either 1k or 10k Resistance Type (available in Rev.3 or later controllers only) causes the Analog Input object to output a value directly reflecting the ohms (Ω) measured at the assigned hardware UI input. The actual ranges supported by the controller UIs for each resistance selection are as follows:

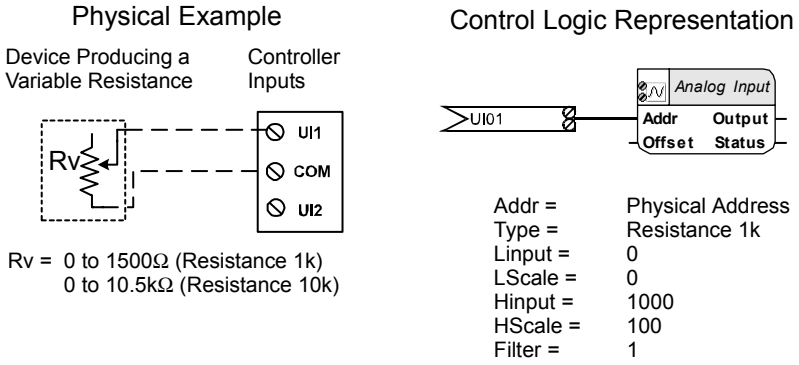
Table–5.22 MN Controller UI Range - 1 k Resistance Selection.

MN Controller	1 kΩ Resistance, Valid Ranges		1 kΩ Resistance, Diagnostic (NA) Limits	
	Low	High	Trip-to-NA	Return-from-NA
Standard	0 Ω	≈1500 Ω	input > ≈1575 Ω	input < ≈1538 Ω
MN 800	0 Ω	≈1500 Ω	input > ≈1538 Ω	input < ≈1500 Ω

Table–5.23 MN controller **UI Range - 10 k Resistance Selection.**

MN Controller	10 k Resistance, Valid Ranges		10 k Resistance, Diagnostic (NA) Limits	
	Low	High	Trip-to-NA	Return-from-NA
Standard	0 Ω	≈10.5k Ω	input > ≈11.02k Ω	input < ≈10.76k Ω
MN 800	0 Ω	≈10.5k Ω	input > ≈10.76k Ω	input < ≈10.50k Ω

Note: High range values above are approximate for both resistance ranges, and may vary a few ohms from controller to controller. For MicroNet standard controllers, the diagnostic limits (trip-to-NA and return-from-NA) are based on a 5% over-range for trip-to-NA and a 2.5% over-range for return-from-NA. For MicroNet MN 800 controllers, the diagnostic limits are based on a 2.5% over-range for trip-to-NA and a 0.0% over-range for return-from-NA.



Figure–5.18 Example Analog Input Object Configured for 1k Resistance Input.

Application Notes

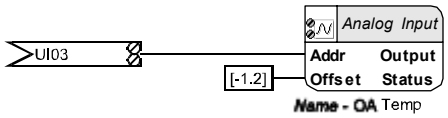
Typical uses for an Analog Input object configured for Resistance include the following applications:

- Actuator feedback, resistance to percent (feeding a Reset object if linear response or a Curve Fit object if non-linear response)
- Non-standard RTD sensors, (feeding a Curve Fit object configured with the necessary resistance-to-temperature sensor data).
- Other custom applications requiring measurement in ohms.

Offset Calibration

Offset calibration defines a value that is added to the calculated output value prior to placing the value at the actual object output. A positive offset value increases the value of the output; a negative offset value decreases the value of the output.

Offset calibration is an input of the Analog Input object. If used, a constant value is typically assigned (connected) as the offset. An unconnected Offset input is equivalent to a not active (NA), causing the Offset Calibration to be set to 0.0 (no offset applied). Offset is useful in calibrating a sensor to a known accurate measuring device, such as a temperature sensor calibrated to a precision thermometer. In the example Analog Input object below Figure-5.19, a -1.2 constant was used to calibrate the sensor reading from a value that (with no offset) was measured to be 1.2 degrees too high.



Figure–5.19 Example AI Object Using Offset Calibration.

Filter

A Filter Constant can be applied to perform an exponential low pass filter which limits the response of the output in relationship to a step change at the input. A filter is typically used to dampen the effects of input change to output change to stabilize a noisy or rapidly changing input signal. The value of the Filter Constant property is limited to values between 0.00 and 1.00. Filter Constant action is shown in Table–5.24 below.

Table–5.24 Filter Constant and Filter Action.

Filter Constant	Filter Action
0.00	No Filter
0.01 (Maximum Filter) through 0.99 (Minimum Filter)	Active Filter Area
1.00 (Default)	No Filter
Not Active (NA)	No Filter

Between any two successive changes at the input, the filter algorithm provides the function as follows:

Output = Previous Output + [Filter Constant (Present Input - Previous Output)]

For example, an Analog Input object for an airflow sensor is observed to have an unstable output near the sensor's low input range, with the object output constantly jumping between 50 and 75 when airflow is holding near 60 CFM. By using a Filter Constant of 0.30, this jumping is minimized.

(At first Input jump from 50 to 75:)

Output = 50.00 + [0.30 (75.0 - 50.00)] or **57.50** (vs. 75.0 with No Filter)

(Input jumps back to 50:)

Output = 57.50 + [0.30 (50.0 - 57.50)] or **55.25** (vs. 50.0 with No Filter)

(Input jumps up to 75:)

Output = 55.25 + [0.30 (75.0 - 55.25)] or **61.18** (vs. 75.0, and so on)

Analog Output

Object Usage: The Analog Output object provides an interface to a physical analog output (AO) point on a controller that produces a 0 to 20 mA current signal. Typically this signal is used to position an analog device such as an actuator or transducer. This point-type object monitors the single assigned input value and determines the proper hardware output signal based upon the assigned input-to-output scaling. Physical devices that operate by using Analog Output objects include:

- Current driven (0 to 20 mA) devices.
- Voltage driven devices, by using a resistor across the device terminals.

Device Support: (all firmware revisions)

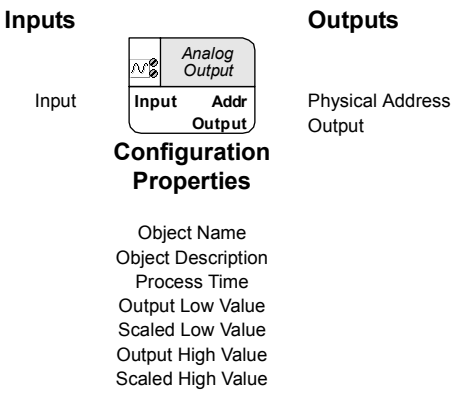
MNL-10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 16 bytes
RAM: 18 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.25 Analog Output Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
LOutput	Output Low Value	Class: Analog - Defines the actual hardware output current (0.0 to 20mA) produced when the input value equals the Scaled Low Value.	4	0.0 to 20.0 mA	

Table–5.25 Analog Output Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
LScale	Scaled Low Value	Class: Analog - Defines the object input value where the hardware output equals the value assigned to the output Low Value (and the Output equals 0.0%)	0	-163.83 to 16383	
HOutput	Output High Value	Class: Analog - Defines the actual hardware output current (0.0 to 20mA) produced when the input value equals the Scaled High Value.	20	0.0 to 20.0 mA	
HScale	Scaled High Value	Class: Analog - Defines the object input value where the hardware output equals the value assigned to the output High Value (and the Output equals 100.0%)	100	-163.68 to 16383	

Table–5.26 Analog Output Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input	Input	Class: Analog - The single input value that is monitored to compute the value at both the object Output and the physical controller (hardware) output.	0.0 to 100.0%	An input with an NA is evaluated the same as zero (0.0) value.

Table–5.27 Analog Output Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Addr	Physical Address	Class: Analog - Defines the physical hardware address (output terminal point on the controller) assigned to the Analog Output object.	Dependent on the controller platform selected.
Output	Output	Class: Analog - The calculated output value, which ranges from 0.0% to 100.0% proportionally as the Input ranges between the Scaled Low Value and Scaled High Value. *Note:* Set to NA if the Input is NA.	0.0 to 100.0%

Applying the Object

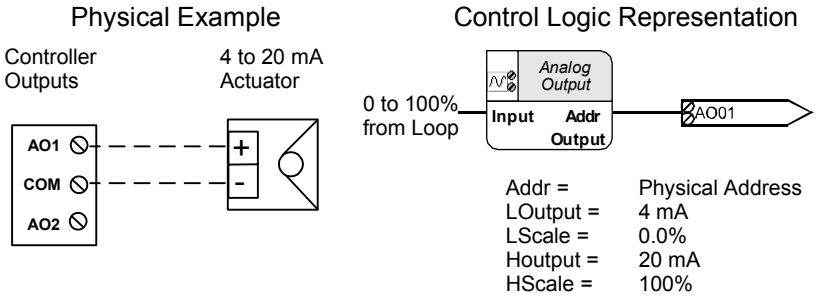
An Analog Output object provides interface to a physical analog output (AO) point on a controller. The hardware output is a 0.0 to 20.0 mA current produced at the AO terminals addressed in the object's Physical Address property. Typical applications include 4 to 20 mA valve and damper actuators, however, voltage-driven devices can also be used by applying a resistor across the device terminals. The object monitors the assigned Input value and determines the proper hardware output signal based on the assigned input-to-output scaling. input-to-output scaling is defined by these four configuration properties:

- Output Low Value
- Output High Value
- Scaled Low Value
- Scaled High Value

Inverted or reverse-acting output signals are achieved by setting the Scaled High Value lower than the Scaled Low Value, or by setting the Output High Value lower than the Output Low Value. Reversing both the scaled and output values simultaneously causes the Analog Output object to provide a direct acting output signal. Whenever the output signals are configured to be reverse-acting, and the Input value is NA, the object's Output will go to NA while the value of the hardware output will go to the value of the Output High Value property.

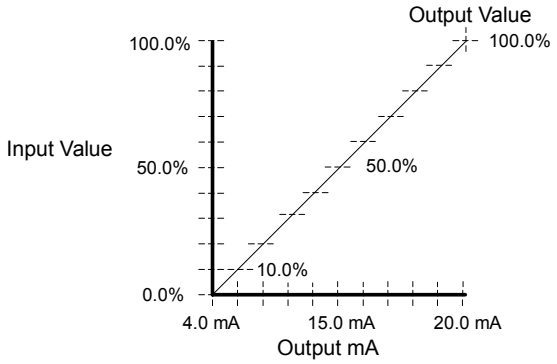
Current Driven Device Example

In this example, the Analog Output object is for a 4 to 20 mA actuator. The Input to the object is a value ranging from 0.0 to 100.0%, which typically comes from a Loop object's output. The input-to-output scaling provides 4.0mA for an Input value of 0.0%, and 20.0mA for an Input of 100.0%.



Figure–5.20 Example Analog Output Object for a 4 to 20 mA Device.

The Output value of the object is a percentage of the Input range as established by the Scaled Low Value and the Scaled High Values.



Figure–5.21 4 to 20 mA Example Relationship of Input Values to Output Values.

Voltage Driven Device Example

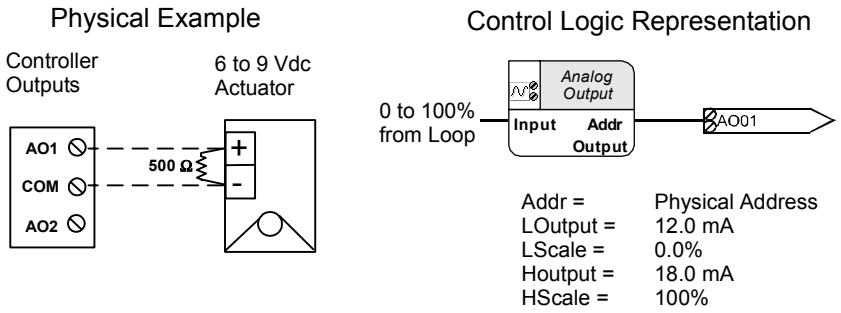
In this example, the Analog Output object is for an actuator that requires a 6 to 9 Vdc signal. Conversion to a voltage output from a current output is accomplished by placing a resistor across the device terminals. An AO output on an I/A Series MicroNet controller can drive a maximum of 550 ohms, producing a maximum output of 11 volts at the full-scale current of 20 mA. Current, voltage, and resistance values relate in this ratio (Ohm's Law):

$$I = \frac{E}{R}$$ where: I = current in amps, E = volts, R = resistance in ohms

If a 500 ohm resistor is used in this example, the output current needed to produce 6 to 9 V is determined by substitution:

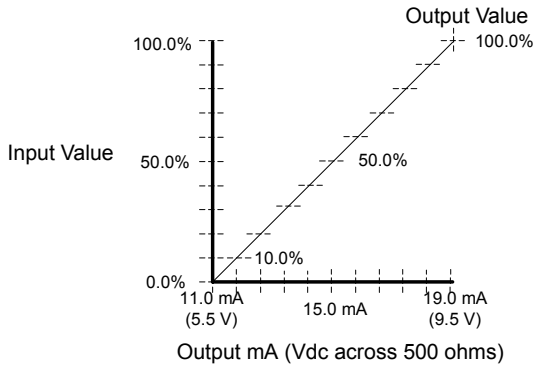
$$I = \frac{6.0\ V}{500\ \Omega} = 0.012 \text{ amps (12 mA)}, \qquad I = \frac{9.0\ V}{500\ \Omega} = 0.018 \text{ amps (18 mA)}$$

The current values become the Output Low Value and Output High Value properties of the Analog Input object, and correspond to the Scaled Low Value and Scaled High Value properties (0.0 and 100.0%) that define the Input to the object.



Figure–5.22 Example Analog Output Object for a 6 to 9 Vdc Device.

The input-to-output scaling above provides an Output of 12.0 mA (6.0 V) when the Input is 0.0%, and 18.0mA (9.0 V) when the Input is 100.0%. Often, a span of 5.5 to 9.5 V is used to ensure complete stroking of this type of actuator. In this case, the Output should be scaled to provide 11.0 mA (5.5 V) when the input is 0.0%, and 19.0mA (9.5 V) when the Input is 100.0%.



Figure–5.23 Example 5.5 to 9.5 Vdc Analog Output, Input Values to Output Values.

Analog Output Priority

Object Usage: Like the Analog Output object, the Analog Output Priority object provides an interface to a physical analog output (AO) point on a controller that produces a 0 to 20 mA current signal to position an analog device such as an actuator or transducer. This point-type object differs from the Analog Output object in that it features four prioritized inputs instead of a single input. The object algorithm chooses the highest valid input and determines the proper hardware output signal based upon assigned input-to-output scaling. Physical devices that operate by using Analog Output Priority objects include:

- Current driven (0 to 20 mA) devices.
- Voltage driven devices, by using a resistor across the device terminals.

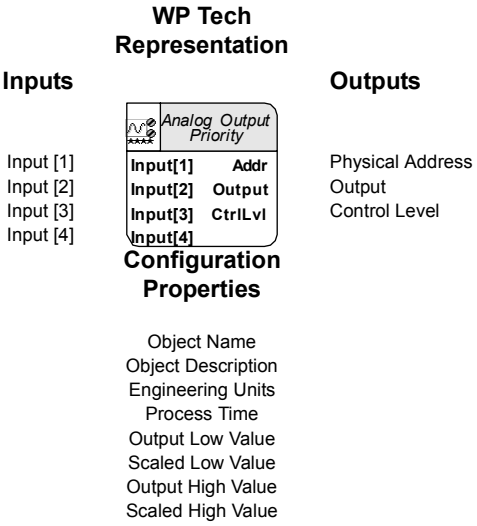
Device Support: (all firmware revisions)

MNL-10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-V2Rxx, -V3Rxx, where xx = V1, V2, or V3
MN 800 series

Memory Requirements: (per object)

EEPROM: 22 bytes
RAM: 26 bytes (standard controllers)
4 bytes (MN 800)

WP Tech Representation



Properties

Table–5.28 Analog Output Priority Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
LOutput	Output Low Value	Class: Analog - Defines the actual hardware output current (0.0 to 20mA) produced when the input value equals the Scaled Low Value.	4	0.0 to 20.0 mA	

Table–5.28 Analog Output Priority Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
LScale	Scaled Low Value	Class: Analog - Defines the object input value where the hardware output equals the value assigned to the output Low Value (and the Output equals 0.0%)	0	-163.83 to 16383	
HOutput	Output High Value	Class: Analog - Defines the actual hardware output current (0.0 to 20mA) produced when the input value equals the Scaled High Value.	20	0.0 to 20.0 mA	
HScale	Scaled High Value	Class: Analog - Defines the object input value where the hardware output equals the value assigned to the output High Value (and the Output equals 100.0%)	100	-163.68 to 16383	

Table–5.29 Analog Output Priority Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input[1]	Input[1]	Class: Analog - The highest priority input value. This input is monitored first to compute the value at both the Output and the physical controller output.	-163.83 to 16,383	A not active (NA) at this input causes the second input to be evaluated for a valid value.
Input[2]	Input[2]	Class: Analog - The second highest priority input value. This input is monitored if Input[1] is NA, and is used to compute the value at both the Output and the physical controller output.	-163.83 to 16,383	A not active (NA) at this input causes the third input to be evaluated for a valid value.
Input[3]	Input[3]	Class: Analog - The third highest priority input value. This input is monitored if Inputs[1] and [2] are both NA, and is used to compute the value at both the Output and the physical controller output.	-163.83 to 16,383	A not active (NA) at this input causes the fourth and last input to be evaluated for a valid value.
Input[4]	Input[4]	Class: Analog - The lowest priority input value. This input is monitored if all other Inputs have a not active (NA), and is used to compute the value at both the Output and the physical controller output.	-163.83 to 16,383	If all inputs including Input[4] have a not active (NA), the Output goes to NA and hardware output goes to the assigned Output Low Value.

Table–5.30 Analog Output Priority Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Addr	Physical Address	Class: Analog - Defines the physical hardware address (output terminal point on the controller) assigned to the Analog Priority object.	Dependent on the controller platform selected.
Output	Output	Class: Analog - The calculated output value, which ranges from 0.0% to 100.0% proportionally as the Input ranges between the Scaled Low Value and Scaled High Value. *Note:* Set to NA if the Input is NA.	0.0 to 100.0%
CtrlLvl	Control Level	Class: Analog - Defines the currently active input by providing the priority number of the related input, that is 1, 2, 3, or 4. If all four inputs have a not active (NA), this output also goes to NA.	1, 2, 3, or 4

Applying the Object

An Analog Output Priority object provides interface to a physical analog output (AO) point on a controller. The hardware output is a 0.0 to 20.0 mA current produced at the AO terminals addressed in the object's Physical Address property. Typical applications include 4 to 20 mA valve and damper actuators, however, voltage-driven devices can also be used by applying a resistor across the device terminals.

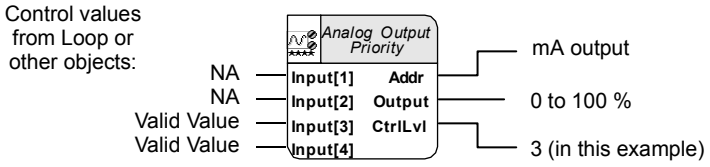
The priority input function allows automatic selection of one of four inputs from the control strategy, based upon priority. The object evaluates the four inputs using a high Input[1] to low Input[4] search and uses the first valid input found and determines the proper hardware (mA) output signal. This output signal is dependent on the assigned input-to-output scaling, defined by these four configuration properties:

- Output Low Value
- Output High Value
- Scaled Low Value
- Scaled High Value

Inverted or reverse acting output signals are achieved by setting the Scaled High Value lower than the Scaled Low Value, or by setting the Output High Value lower than the Output Low Value. Reversing both the scaled and output values simultaneously causes the Analog Output object to provide a direct acting output signal.

Priority Inputs and Values

Input[1] is the highest priority input, and is always evaluated first on each scan of the inputs. Any valid value present on Input[1] becomes the Input to the object, regardless of the state of the other inputs. A valid value is any numeric value besides a not active [NA].



Figure–5.24 Input[3] as the Current Active Input.

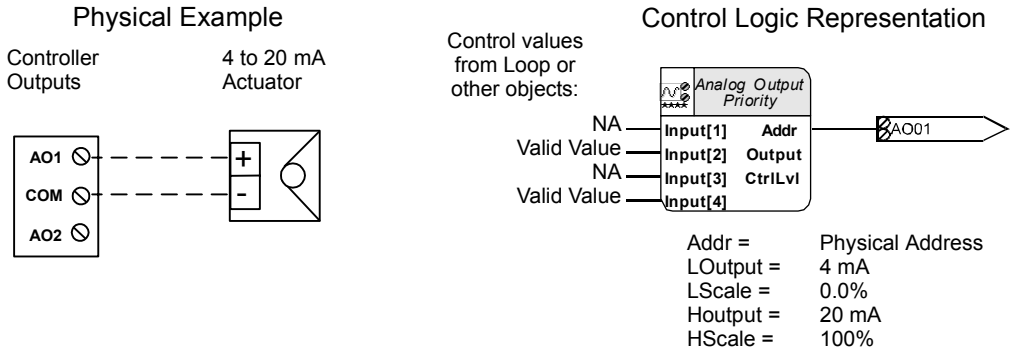
If Input[1] has an NA, then Input[2] is evaluated in the same manner. This priority scan continues only if Input[2] also has an NA, at which point Input[3] is evaluated, and if Input[3] also has an NA, to lastly evaluate Input[4]. If Input[4] also has an NA, then the Output goes to Not Active and the hardware output goes to the assigned Output Low Value.

Typically, input values are within the range of the object's input scaling, that is, between the Input Low Scale and Input High Scale. However, any value outside this range is evaluated as either the value of the Input Low Scale or the value of the Input High Scale.

For example, a typical object has an Input Low Scale of 0.0 and an Input High Scale of 100.0. Input values typically fall within this range. In this example, if the value of an input is above 100.0, for example, 165.0, it is evaluated by the object as 100.0. Likewise, a negative value such as - 56.7 would be evaluated by the object as 0.0.

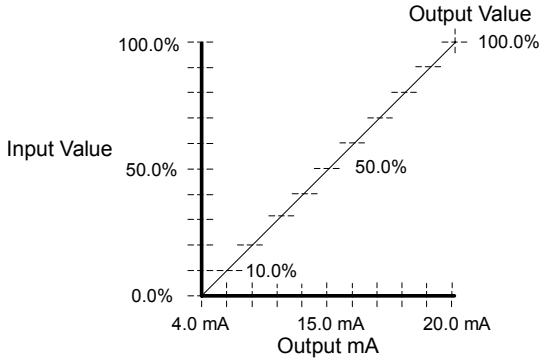
Current Driven Device Example

In this example, the Analog Output Priority object is for a 4 to 20 mA actuator. Inputs[1] through [4] receive various control values from Loop objects or other parts of the control application. At this moment, values at Inputs[2] and [4] range between 0.0 to 100.0%, while Inputs[3] and [4] indicate invalid, or not active (NA) conditions. The object selects the highest priority valid input (Input[2]) as the value used for the algorithm. The input-to-output scaling provides a hardware output of 4.0mA for an Input value of 0.0% and 20.0mA for an Input of 100.0%. The Control Level output indicates the valid input used, in this case, a value of 2.



Figure–5.25 Example Analog Output Object for a 4 to 20 mA Device.

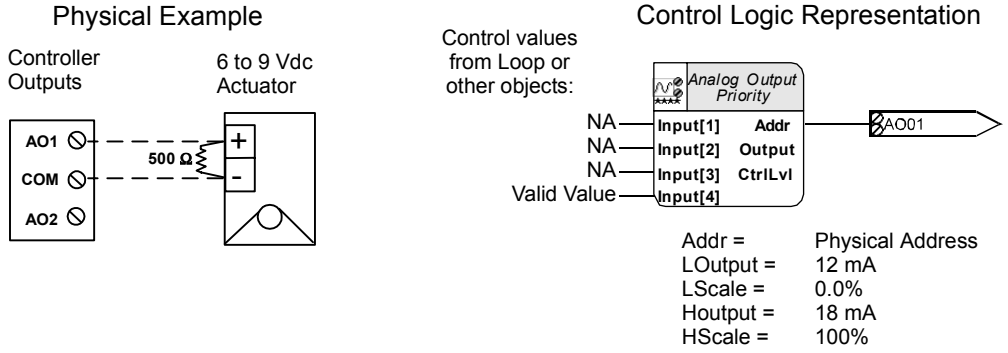
The Output value of the object is a 0.0 to 100.0 value, representing the percentage of the Input range as established by the Scaled Low Value and the Scaled High Values as shown below.



Figure–5.26 Output Value as a Percentage of Example 0 to 100% Input Range.

Voltage Driven Device Example

In this example, the Analog Output object is for a 6 to 9 Vdc actuator. Inputs[1] through [4] receive various control values from Loop objects or other parts of the control application. At this moment, only Input[4] has a valid value that ranges between 0.0 to 100.0%, while Inputs[2], [3], and [4] indicate invalid, or not active (NA) conditions. The object selects the highest priority valid input (Input[4]) as the value used for the algorithm.



Figure–5.27 Example Analog Output Object for a 6 to 9 Vdc Device.

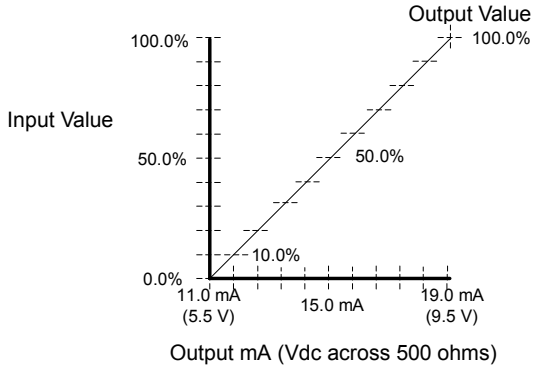
Conversion to a voltage output from a current output is accomplished by placing a resistor across the device terminals. An AO output on an I/A Series MicroNet controller can drive a maximum of 550 ohms, producing a maximum output of 11 volts at the full-scale current of 20 mA. Current, voltage, and resistance values relate in this ratio (Ohm's Law):

$$I = \frac{E}{R}$$ where: I = current in amps, E = volts, R = resistance in ohms

If a 500 ohm resistor is used in this example, the output current needed to produce 6 to 9 V is determined by substitution:

$$I = \frac{6.0\ V}{500\,\Omega} = 0.012 \text{ amps (12 mA)}, \qquad I = \frac{9.0\ V}{500\,\Omega} = 0.018 \text{ amps (18 mA)}$$

This input-to-output scaling provides an Output of 12.0 mA (6.0 V) when the Input is 0.0%, and 18.0mA (9.0 V) when the Input is 100.0%. Often, a span of 5.5 to 9.5 V is used to ensure complete stroking of this type of actuator. In this case, the Output is scaled to provide 11.0 mA (5.5 V) when the input is 0.0%, and 19.0mA (9.5 V) when the Input is 100.0%.



Figure–5.28 Example Input Value, Output mA, and Output Value Relationship.

AND / AND

Object Usage: The AND / AND object is a three-input logic object for use with OFF and ON digital values (DV). The output of the object is a digital ON only when all valid inputs are in a digital ON state. Any input in a digital OFF state results in an output of digital OFF. An unconnected input is considered invalid or not active (NA), and is ignored in the object's algorithm. If all inputs are NA, the output is set to NA.

Output = In1 AND In2 AND In3

Device Support: (all firmware revisions)

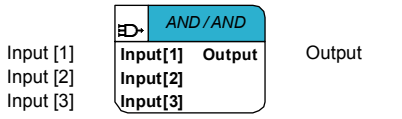
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

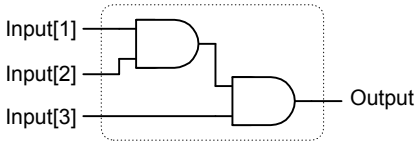
WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

Logic



AND / AND

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Digital Logic Objects

Object Name	Digital Object Algorithm (all are three-input unless noted)
AND / AND	In1 AND In2 AND In3
AND / OR	(In1 AND In2) OR In3
Clocked SR	Clocked Set-Reset Flip-Flop Logic
EXOR	Two-input, Exclusive OR
Latch	Digital Sample and Hold or Latch
OR / AND	(In1 OR In2) AND In3
OR / OR	In1 OR In2 OR In3
SR Flip-Flop	Two-input, Set-Reset Flip-Flop Logic

Properties

Table–5.31 AND / AND Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.32 AND / AND Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Digital - The first input evaluated for an ON. If OFF, the output is set to OFF. A not active (NA) is ignored. **In1** AND In2 AND In3	—	See the Truth Table for all possible input combinations and corresponding outputs.
Input[2]	Input [2]	Class: Digital - The second input evaluated for an ON. If OFF, the output is set to OFF. A not active (NA) is ignored. In1 AND **In2** AND In3	—	See the Truth Table for all possible input combinations and corresponding outputs.
Input[3]	Input [3]	Class: Digital - The third input evaluated for an ON. If OFF, the output is set to OFF. If not active (NA), the input is ignored unless all inputs are NA, in which case the output is also set to NA. In1 AND In2 AND **In3**	—	See the Truth Table for all possible input combinations and corresponding outputs.

Table–5.33 AND / AND Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The output indicates the result of the logic algorithm. If not active (NA) is present at all three inputs, the output is set to NA.	OFF (0.0) ON (100.0)

Applying the Object

The AND / AND object is similar to other three-input logic objects, which also process OFF and ON digital values (DV) and produce an DV output. The object's algorithm is unique in the use of two logical AND operators:

In1 AND In2 AND In3

The object logic calls for all Inputs with valid digital values to be ON before the Output is set to ON, otherwise the Output is OFF. If an Input is not active (NA) it is invalid (ignored), allowing less than three Inputs to be evaluated for an ON state. If all Inputs are NA, the Output is set to NA.

Table–5.34 provides the result of all AND / AND input combinations, including NA inputs.

Table–5.34 Truth Table for AND / AND Object.

Input[1]	Input[2]	Input[3]	Output
OFF	OFF	OFF	OFF
OFF	OFF	ON	OFF
OFF	OFF	NA	OFF
OFF	ON	OFF	OFF
OFF	ON	ON	OFF
OFF	ON	NA	OFF
OFF	NA	OFF	OFF
OFF	NA	ON	OFF
OFF	NA	NA	OFF
ON	OFF	OFF	OFF
ON	OFF	ON	OFF
ON	OFF	NA	OFF
ON	ON	OFF	OFF
ON	ON	ON	ON
ON	ON	NA	ON
ON	NA	OFF	OFF
ON	NA	ON	ON
ON	NA	NA	ON
NA	OFF	OFF	OFF
NA	OFF	ON	OFF
NA	OFF	NA	OFF
NA	ON	OFF	OFF
NA	ON	ON	ON
NA	ON	NA	ON
NA	NA	OFF	OFF
NA	NA	ON	ON
NA	NA	NA	NA

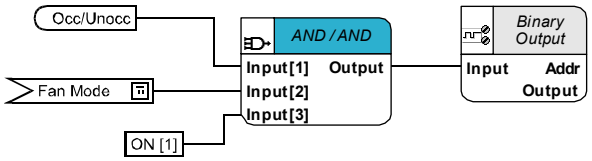
As with other logic objects, Inputs to this object are typically digital values from a connection to another object's output, represented numerically with a 0 (OFF) or 100 (ON). However, Inputs can also process analog values or a value from an assigned constant. An Input with an analog value or constant is evaluated as OFF if the value is negative or equal to zero (0.0), or ON if the Input has any positive value greater than zero.

Example Application

In this example, the AND / AND object is used to send an ON or OFF to a Binary Output object for a fan. Inputs to the object are as follows:

- Input [1] represents an ON or OFF schedule command.
- Input[2] is a ON or OFF Fan signal from a MicroNet Sensor.
- Input[3] is a constant value 1, always representing an ON.

Due to the constant on Input[3], this AND / AND object only requires each of the first two Inputs to change to ON before the Output changes to ON, starting the object for the fan.



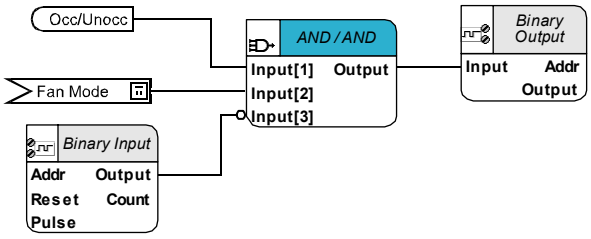
Figure–5.29 Example AND / AND Object.

In this case, the object would behave the same way if Input[3] was left unconnected, acting as not active (NA). However, attaching a constant value provides a convenient way to override the output OFF when online with the controller (using the" Write to RAM" function available with constant tags).

Inverted Inputs

As with all control logic objects, each Input on an AND / AND object can be configured as inverted to test for an opposite state input. Inversion of inputs is particularly useful with logic objects.

With an AND / AND object, each inverted input tests for a digital OFF, rather than a digital ON as with a normal (non-inverted) input. An example of an inverted input is shown below in a variation of the previous example. Input[3] is no longer connected to a constant 1 (ON), but is looking at a Binary Input object for an OFF before the object Output turns on the fan. The Binary Input object is for a low-limit switch, where an ON indicates temperature is too low for fan operation.



Figure–5.30 Example AND / AND Object Using an Inverted Input.

AND / OR

Object Usage: The AND / OR object is a three-input logic object for use with OFF and ON digital values (DV). The output of the object is a digital ON whenever the first two valid inputs (Input[1] and Input[2]) are both in a digital ON state, or if Input[3] is in a digital ON state. Other Input combinations result in an output of OFF. An unconnected input is considered invalid or not active (NA), and is ignored in the object's algorithm. If all inputs are NA, the output is set to NA.

Output = (In1 AND In2) OR In3

Device Support: (all firmware revisions)

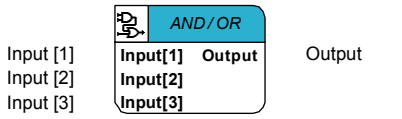
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

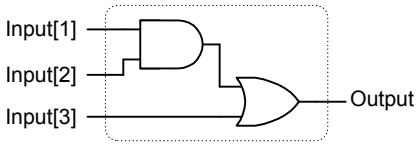
WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

Logic



AND / OR

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Digital Logic Objects

Object Name	Digital Object Algorithm (all are three-input unless noted)
AND / AND	In1 AND In2 AND In3
AND / OR	(In1 AND In2) OR In3
Clocked SR	Clocked Set-Reset Flip-Flop Logic
EXOR	Two-input, Exclusive OR
Latch	Digital Sample and Hold or Latch
OR / AND	(In1 OR In2) AND In3
OR / OR	In1 OR In2 OR In3
SR Flip-Flop	Two-input, Set-Reset Flip-Flop Logic

Properties

Table–5.35 AND / OR Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.36 AND / OR Object Input Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Input[1]	Input [1]	Class: Digital The first input evaluated for an ON. If OFF, the third input is evaluated. A not active (NA) is ignored. **(In1** AND In2) OR In3	—	—	See the Truth Table for all possible input combinations and corresponding outputs.
Input[2]	Input [2]	Class: Digital The second input evaluated for an ON. If OFF, the third input is evaluated. A not active (NA) is ignored. (In1 AND **In2**) OR In3	—	—	See the Truth Table for all possible input combinations and corresponding outputs.
Input[3]	Input [3]	Class: Digital The third input evaluated for an ON. If OFF, the output is set to OFF unless both Inputs[1] and [2] are ON. If all inputs are NA, the output is set to NA. (In1 AND In2) OR **In3**	—	—	See the Truth Table for all possible input combinations and corresponding outputs.

Table–5.37 AND / OR Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital The output indicates the result of the logic algorithm. If not active (NA) is present at all three inputs, the output is set to NA.	OFF (0) ON (100)

Applying the Object

The AND / OR object is similar to other three-input logic objects, which also process OFF and ON digital values (DV) and produce an DV output. The object's algorithm uses this logic:

(In1 AND In2) OR In3

This logic requires either of these Input states before the Output is ON:

- both Inputs[1] and [2] to be ON, or
- Input[3] to be ON.

Otherwise, the Output is OFF. An unconnected Input is considered not active (NA) and is invalid (ignored), allowing less than three Inputs to be evaluated. If all Inputs are NA, the Output is set to NA.

Table–5.38 below provides all AND / OR object input/output combinations.

Table–5.38 Truth Table for AND / OR Object

Input[1]	Input[2]	Input[3]	Output
OFF	OFF	OFF	OFF
OFF	OFF	ON	ON
OFF	OFF	NA	OFF
OFF	ON	OFF	OFF
OFF	ON	ON	ON
OFF	ON	NA	OFF
OFF	NA	OFF	OFF
OFF	NA	ON	ON
OFF	NA	NA	OFF
ON	OFF	OFF	OFF
ON	OFF	ON	ON
ON	OFF	NA	OFF
ON	ON	OFF	ON
ON	ON	ON	ON
ON	ON	NA	ON
ON	NA	OFF	ON
ON	NA	ON	ON
ON	NA	NA	ON
NA	OFF	OFF	OFF
NA	OFF	ON	ON
NA	OFF	NA	OFF
NA	ON	OFF	ON
NA	ON	ON	ON
NA	ON	NA	ON
NA	NA	OFF	OFF
NA	NA	ON	ON
NA	NA	NA	NA

As with other logic objects, Inputs to this object are typically digital values from a connection to another object's output, represented numerically with a 0 (OFF) or 100 (ON). However, Inputs can also process analog values or a value from an assigned constant. An Input with an analog value or constant is evaluated as OFF if the value is equal to or less than zero (0.0), or ON if the Input has any positive value greater than zero.

Average

Object Usage: The Average object is a three-input math object for use with analog values (AV). This object calculates the average value of all the valid inputs applied.

Output = Average (AV1 , AV2 , AV3)

Only one valid input is required to produce a valid output.

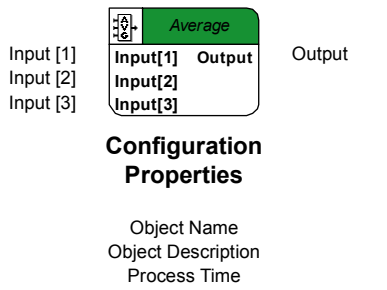
Device Support: (all firmware revisions)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3
MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.39 Average Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.40 Average Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The first value used in the average calculation. A not active (NA) is not evaluated.	-163.83 to 16383	If not active (NA) is present at all Inputs, the output is set to NA.
Input[2]	Input [2]	Class: Analog - The second value used in the average calculation. A not active (NA) is not evaluated.	-163.83 to 16383	
Input[3]	Input [3]	Class: Analog - The third value used in the average calculation. A not active (NA) is not evaluated.	-163.83 to 16383	

Table–5.41 Average Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output is the average of all valid Inputs. If not active (NA) is present at all Inputs, the output is set to NA.	-163.83 to 16383

Applying the Object

The Average object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. The equation specific to the Average object is:

(AV1 + AV2 + AV3) ÷ *n* where *n* is the number of valid Inputs.

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.00 for OFF or 100.00 for ON), or not active (NA).

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. Table–5.42 shows how NA inputs affect the output of the Average object.

Table–5.42 Average Object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	Average (AV1, AV2, AV3)
AV1	AV2	NA	Average (AV1, AV2)
NA	AV2	AV3	Average (AV2, AV3)
AV1	NA	AV3	Average (AV1, AV3)
AV1	NA	NA	AV1
NA	AV2	NA	AV2
NA	NA	AV3	AV3
NA	NA	NA	NA

Binary Alarm

Object Usage: The Binary Alarm object provides for detection and annunciation of ON/OFF type alarms and return-from-alarm conditions within an application. The user can specify the alarm state and an alarm delay time. Alarm and return from alarm conditions are indicated at the object output and can be stored in the controller's local alarm buffer as alarm message ID numbers, which in turn can be viewed at the controller's MicroNet sensor (MN-S3, S4, S4-FCS, or S5).

Device Support: (all firmware revisions)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4

MNL-11Rxx, -13Rxx
where xx = F2 or F3

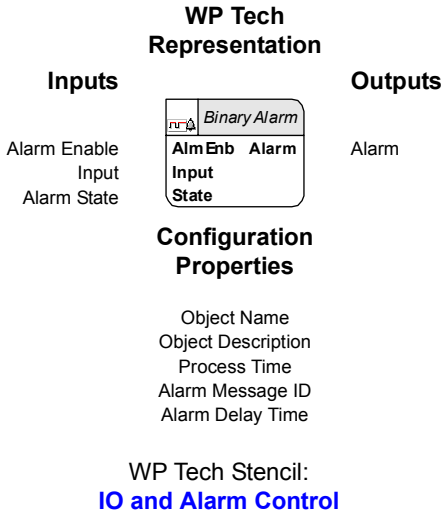
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 16 bytes
RAM: 22 bytes (standard controllers)
6 bytes (MN 800)

WP Tech Representation



Properties

Table–5.43 Binary Alarm Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.43 Binary Alarm Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
MsgID	Alarm Message ID	Class: Analog - A user-assigned alarm message ID associated with the alarm condition within the application. A value of zero indicates that a message ID is not assigned. A not active (NA) or value outside the defined range causes the Alarm Message ID be evaluated as zero.	0	1 to 127	(pre-Rev.3 controllers): A return from alarm condition adds 128 to the assigned Alarm Message ID, writing a value between 129 and 255 into the local alarm buffer.
ADlyTm	Alarm Delay Time	Class: Analog - Defines the length of time (in seconds) that the Binary Alarm object must: • Be in an alarm condition before generating an alarm. • Return to a non-alarm condition before generating a return from alarm. An alarm or return from alarm includes generation of network alarm messages and an update of the object alarm output.	0	0.0 to 10,000 seconds	A not active (NA) causes the Alarm Delay Time value to be set to 0.0 seconds.

Table–5.44 Binary Alarm Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
AlmEnb	Alarm Enable	Class: Digital - An input of not active (NA) or Digital ON enables the Binary Alarm function. An input value of Digital OFF causes the algorithm to: • Hold all outputs at their previous state. • Reset the Alarm Delay Time timeouts. • Disable alarm reporting for this object. ***Note:*** If the Binary Alarm object is in an *active alarm state* when an input value of Digital OFF is received, the object *will remain* in the alarm state. Be sure the object *is removed* from the active alarm state before disabling the Binary Alarm function.	—	For each input this is typically a digital value, that is, OFF (0.0) or ON (100.0). Analog values are evaluated as: Negative (<0) = OFF Positive (>0) = ON
Input	Input	Class: Digital - The input compared against the state value assigned to the Alarm State input to determine the binary alarm condition. A not active (NA) at this input causes the algorithm to: • Disable alarm reporting for this object. • Hold all outputs at their previous state. • Reset the Alarm Delay Time timeouts.	—	
State	Alarm State	Class: Digital - Defines the binary alarm activation (trip state) that the Input is continuously monitored and compared against. A not active (NA) causes the binary alarm function to be inactive.	—	

Table–5.45 Binary Alarm Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Alarm	Alarm	Class: Digital - This output is set to ON whenever the Binary Alarm algorithm has determined an alarm condition. An OFF indicates that an alarm condition does not exist.	Normal is OFF (0.0) Alarm is ON (100.0)

Applying the Object

The Binary Alarm object monitors the digital value on its Input and compares it to the digital value assigned to the Alarm State. If the digital values match, an alarm sequence begins. An alarm sequence results in an alarm only if the Input remains matched to the Alarm State for a period of time exceeding the assigned Alarm Delay Time. The following table shows the result of all Alarm State and Input conditions:

Table–5.46 Binary Alarm Object Alarm State and Input Combinations.

Alarm State	Input	Output	Alarm Condition
Digital OFF	Digital OFF	Digital ON	Alarm
Digital OFF	Digital ON	Digital OFF	Normal
Digital ON	Digital OFF	Digital OFF	Normal
Digital ON	Digital ON	Digital ON	Alarm

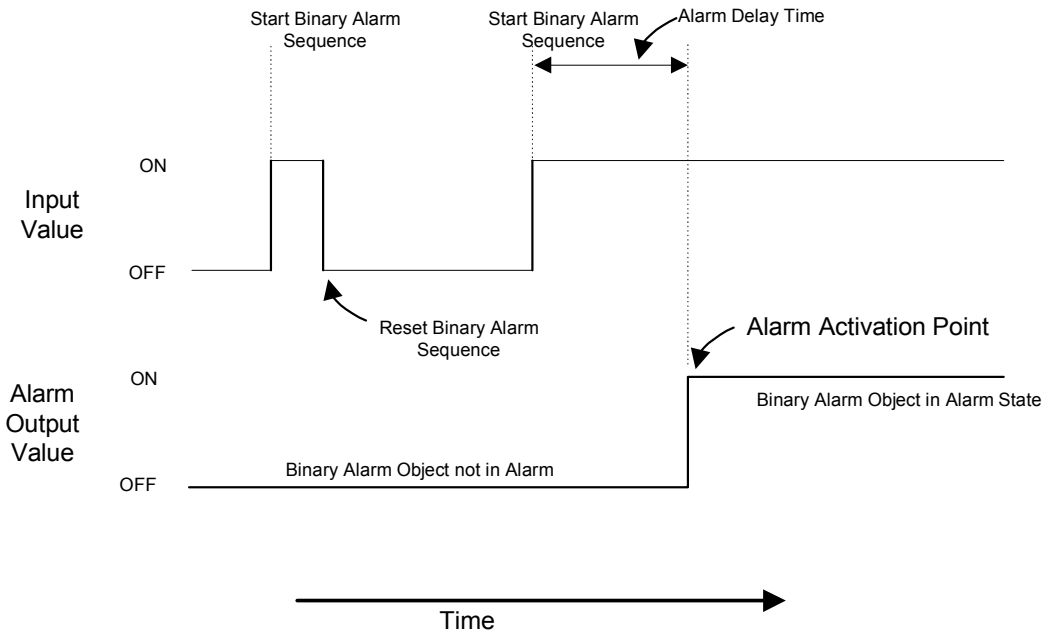
Note: Output and Condition is after completion of Alarm Delay.

Alarm Sequence and Alarm Activation

A binary alarm sequence is initiated whenever the Input matches the assigned Alarm State.

- If the Input remains in this state for a period of time greater than the assigned Alarm Delay Time, an alarm is activated, and the object output Alarm is set to a digital ON.
- If the Input returns to a normal state prior to expiration of the Alarm Delay Time, the binary alarm sequence is reset.

A graphical representation of an alarm sequence and alarm activation is shown in Figure–5.31 below. In this example the Alarm State = Digital ON.



Figure–5.31 Binary Alarm Sequence and Alarm Activation.

Binary Alarm

Activation of a binary alarm initiates the following events:

- The alarm is indicated at the object's output.
- The alarm may be sent to the controller's alarm buffer.

Binary Alarm Object Output

The Binary Alarm object indicates the binary alarm condition by setting the Alarm output to a Digital ON. An Alarm output of Digital OFF indicates that a binary alarm condition does not exist.

Local Alarm Buffer

Each MicroNet controller has its own local alarm buffer. This local alarm buffer contains the last four reported alarm message ID's within the controller, which can be reviewed at the LCD screen of the MicroNet sensor connected to the controller (MN-S3xx, S4xx, S4xx-FCS, or S5xx models).

The Binary Alarm object reports the alarm to the local alarm buffer by sending it the corresponding assigned Alarm Message ID. The valid range for an Alarm Message ID is between 1 and 128.

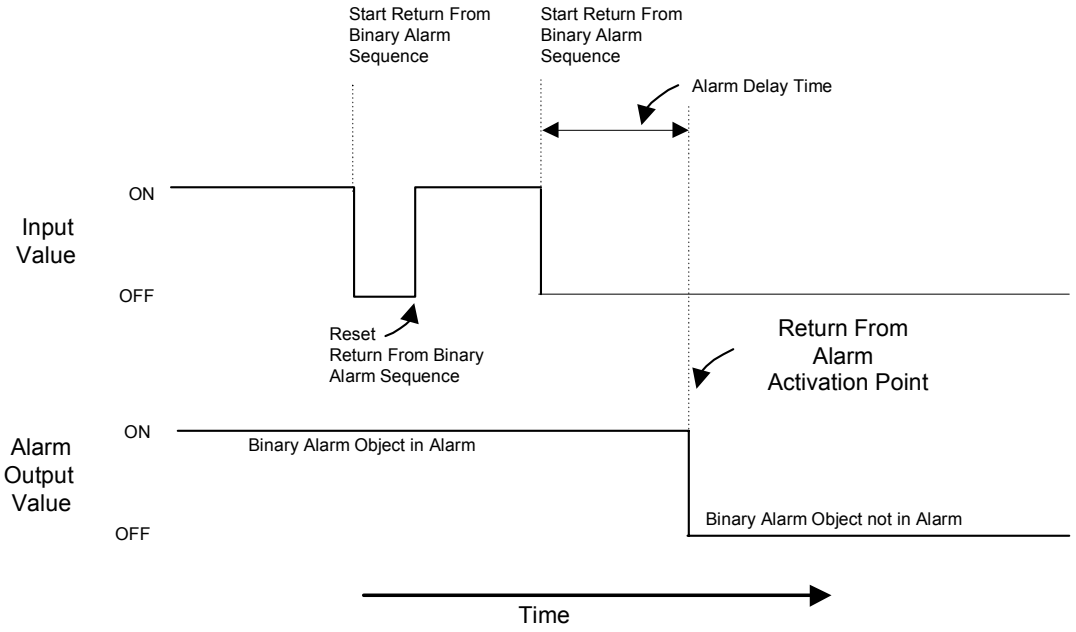
An Alarm Message ID of zero, not active (NA), or a value outside the defined range indicates that a message ID is not assigned. In this case, the Alarm Message ID is not sent to the local alarm buffer.

Return from Binary Alarm Sequence and Activation

A return from binary alarm sequence is initiated whenever an alarm is active and the Input no longer matches the assigned Alarm State.

- If the Input remains in this state for a period of time greater than the assigned Alarm Delay Time, a return from alarm is activated, and the object output Alarm is set to a digital OFF.
- If the Input returns to the alarm state prior to expiration of the Alarm Delay Time, the return from binary alarm sequence is reset.

A graphical representation of a return from alarm sequence and return from alarm activation is shown in the following diagram Figure-5.32.
In this example the Alarm State = Digital ON.



Figure–5.32 Example Return from Binary Alarm Sequence and Activation.

A return from a binary alarm initiates the following events:

- The return from alarm is indicated at the object's outputs.
- The return from alarm may be sent to the controller's alarm buffer.

Binary Alarm Object Output

The Binary Alarm object indicates a return from binary alarm condition by setting the Alarm output to a Digital OFF. An Alarm output of Digital OFF indicates that an alarm condition no longer exists.

Local Alarm Buffer

As described previously, each controller has a local alarm buffer that holds the last four reported alarm message ID's within the controller, which can be reviewed by devices such as MN-S3xx, S4xx, S4xx-FCS, and S5xx sensors.

Note: Rev.3 or higher controllers store only "active" alarms, with Alarm Message IDs in the range of 1 to 127. The next paragraph applies only to controllers with pre-Rev.3 firmware.

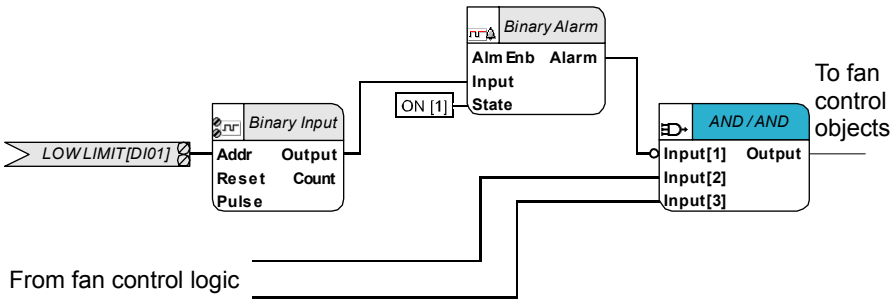
In a pre-Rev.3 controller, when the Binary Alarm object has a return-from-alarm condition, it automatically adds 128 to the corresponding assigned High or Low Alarm Message ID value. This incremented value is then stored in the local alarm buffer as a Return from Alarm Message ID. This makes the valid range of values between 129 and 255 for both a Return from Alarm Message ID.

An Alarm Message ID of zero, not active (NA), or a value outside the defined range indicates that a message ID is not assigned. In this case, Alarm Message IDs are not sent to the local alarm buffer.

Example Applications

Constant Alarm State

The Binary Alarm object in this example is used to monitor the status of a duct-mounted low-limit temperature switch. An alarm delay time of 10 seconds is used. The Binary Alarm object Input connects to the output of the Binary Input object used for the low-limit temperature switch, and a constant value 1 (ON) is assigned to the Alarm State.



Figure–5.33 Binary Alarm Object with Constant Alarm State.

In this example, the output of the Binary Alarm object is fed to an inverted input of an AND / AND logic object, combined with other digital fan control logic (not shown) in this application. If the low-limit switch trips and holds for 10 or more seconds, the Binary Alarm object sets its Alarm output ON, which is read as an OFF on the inverted input of the AND / AND object. This produces an OFF on the output of the AND / AND object, resulting in a fan shutdown sequence.

Alarm Message ID: The programmer assigns a value of 10 to the property Alarm Message ID number. This non-zero value enables storage of an alarm to the controller's local alarm buffer.

If this low-limit alarm occurs, the Alarm Message ID of "010" can be seen from the "ALr" portion of the Diagnostics screens, accessible from the controller's MN-S3xx, S4xx, S4xx-FCS, or S5xx model MicroNet sensor.

Note: Diagnostics screens of an MN-S3xx, S4xx, S4xx-FCS or S5xx sensor are brought up by pressing and holding the sensor's entire Up/Down Key for five seconds. Two more Up presses of the Up/Down Key produce the local object alarm buffer, indicated by the flashing "ALr" message followed by two flashes for each of the four possible stored Alarm Message IDs.

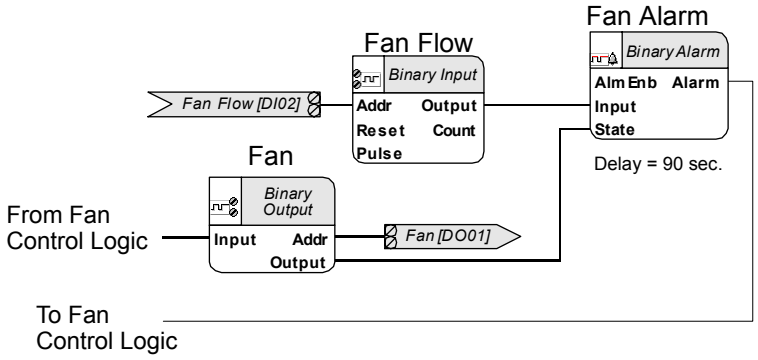
Controllers with Rev.3 or later firmware store only message IDs for "active" alarms. An Alarm Message ID is cleared from the buffer on return from alarm. Return-from-alarm message IDs (those incremented by 128) are stored only in controllers with earlier firmware (MNL-10Rx1, -20Rx1, -VxR1).

Variable Alarm State

The Alarm State property exists as an input to the Binary Alarm object. This allows the alarm state to be easily changed by the application so that the Binary Alarm object can test the Input for different states, depending on the current need. In the example in Figure–5.34, a Binary Alarm object is used to report an alarm condition of a fan, referencing the output of a Binary Input object for a proof-of-flow switch. The Alarm State is determined by the output of the Binary Output object for the fan, indicating the current ON or OFF command for the fan. Switching of Alarm State with a fan command change allows a fan alarm when either:

- The fan is commanded ON, but proof-of-flow is not detected.
- The fan is commanded OFF, but proof-of-flow is detected.

In this case, the Alarm Delay Time is set to a value high enough to avoid a false alarm after a fan command change (90 seconds). As an example, this delay time is necessary following a fan OFF transition, when the fan flow feedback circuit continues to report ON for some time as the fan flow winds down.



Figure–5.34 Binary Alarm Object Example Using a Variable Alarm State.

Binary Encoder

Object Usage: The Binary Encoder object converts a linear sequence of digital input signals to a binary encoded sequence of digital outputs. Binary encoding allows more combinations of output levels with fewer output loads than needed with linear sequenced outputs. The typical application is to pair this object with a Sequence (6) object (page 481) to provide stage control for three electric heat loads sized to provide six unique output levels of control. The Binary Encoder algorithm monitors the linear progression on the six digital inputs and delivers an equivalent binary encoded output combination to the three digital outputs.

Device Support: (all firmware revisions)

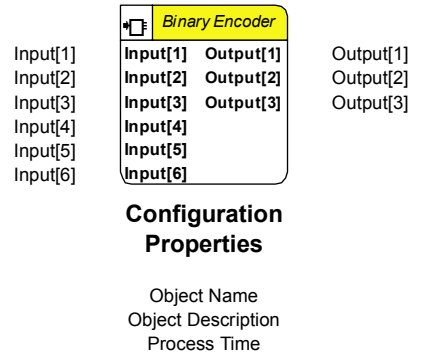
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 16 bytes
RAM: 22 bytes (standard controllers)
6 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.47 Binary Encoder Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.48 Binary Encoder Object Input Properties.

Abbrev.	Name	Class / Description	Default	**Range / Selection**	Notes
Input[1]	Input[1]	Class: Digital - Defines the first input in the linear sequence to be evaluated for an ON.	—	—	An OFF or an NA at this input causes this and all higher inputs to be ignored, and for all outputs to be OFF.
Input[2]	Input[2]	Class: Digital - Defines the second input in the linear sequence to be evaluated for an ON.	—	—	An OFF or an NA at this input causes this and all higher inputs to be ignored.
Input[3]	Input[3]	Class: Digital - Defines the third input in the linear sequence to be evaluated for an ON.	—	—	An OFF or an NA at this input causes this and all higher inputs to be ignored.
Input[4]	Input[4]	Class: Digital - Defines the fourth input in the linear sequence to be evaluated for an ON.	—	—	An OFF or an NA at this input causes this and all higher inputs to be ignored.
Input[5]	Input[5]	Class: Digital - Defines the fifth input in the linear sequence to be evaluated for an ON.	—	—	An OFF or an NA at this input causes this and all higher inputs to be ignored.
Input[6]	Input[6]	Class: Digital - Defines the sixth input in the linear sequence to be evaluated for an ON.	—	—	An OFF or an NA at this input causes this and all higher inputs to be ignored.

Table–5.49 Binary Encoder Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output[1]	Class: Digital - Represents the least significant bit (2n where n = 0) in the overall binary encoded output, using a weight of 1. This output is ON if the linear input sequence ends on Inputs[1], [3], or [5].	OFF (0.0) ON (100.0)
Output[2]	Output[2]	Class: Digital - Represents the next significant bit (2n where n = 1) in the overall binary encoded output, using a weight of 2. This output is ON if the linear input sequence ends on Inputs[2], [3], or [6].	OFF (0.0) ON (100.0)
Output[3]	Output[3]	Class: Digital - Represents the most significant bit (2n where n = 2) in the overall binary encoded output, using a weight of 4. This output is ON if the linear input sequence ends on Inputs[4], [5], or [6].	OFF (0.0) ON (100.0)

Applying the Object

The Binary Encoder object is typically matched with a Sequence (6) object for load staging control in an electric heat application. The Sequence (6) object is typically the input half of this two-object logic combination, which linearly sequences ON a set of digital logic outputs in proportion to a 0 to 100% input signal (typically sent from a heating loop).

Binary Encoded Output

The Binary Encoder object is the output half of this logic controller. This object receives the linearly sequenced digital signals from the Sequence object on Inputs[1] through [6] and produces the binary encoded equivalent on Outputs[1] through [3], as shown in the following chart Figure-5.50.

Table–5.50 Binary Encoder Object Input to Output Truth Table.

Linear Sequenced Inputs						Binary Encoded Outputs		
Input[1]	**Input[2]**	**Input[3]**	**Input[4]**	**Input[5]**	**Input[6]**	**Output[3] MSB[a]**	**Output[2]**	**Output[1] LSB1**
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON
ON	ON	OFF	OFF	OFF	OFF	OFF	ON	OFF
ON	ON	ON	OFF	OFF	OFF	OFF	ON	ON
ON	ON	ON	ON	OFF	OFF	ON	OFF	OFF
ON	ON	ON	ON	ON	OFF	ON	OFF	ON
ON	ON	ON	ON	ON	ON	ON	ON	OFF

a. Note: MSB is Most Significant Bit, and LSB is Least Significant Bit.

Outputs[1] to [3] typically control three Binary Output objects, each of which switches ON or OFF a uniquely sized load. Load sizing assumes a binary weighted proportion (factors 1, 2, and 4), for example, electrical loads of 3kW, 6kW, and 12kW. This load sizing allows an even gradient heat output across the various input to output stage combinations.

Out of Sequence Inputs or NA Inputs

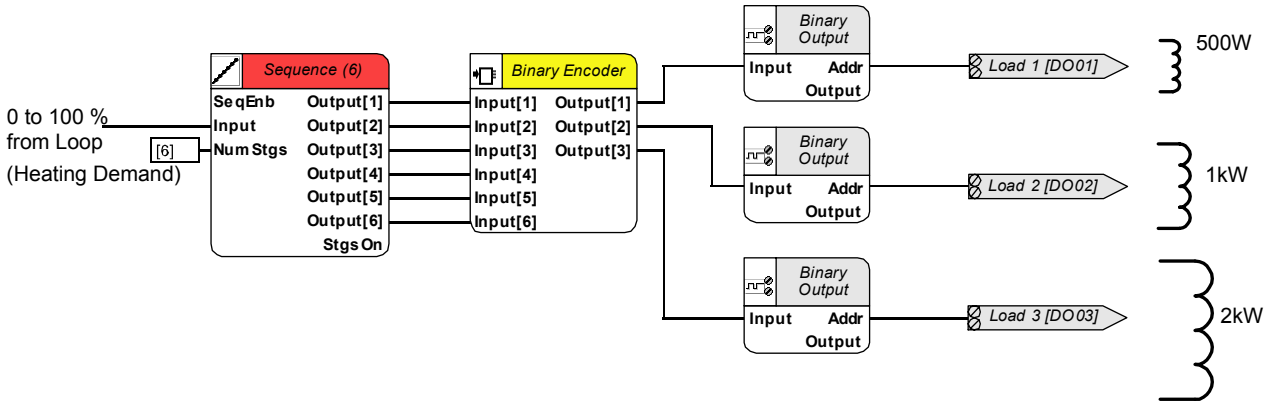
The Binary Encoder object continuously scans Inputs[1] through [6] for a digital ON in a strictly low to high linear sequence, from Input[1] to [6]. If any Input is found in an OFF or NA condition, evaluation of that Input (and all higher Inputs) ends on that input scan, and Outputs[1] to [3] are set based on the previous Inputs evaluated. Examples and effects of out of sequence OFFs or NAs are shown below Figure-5.51.

Table–5.51 Example Effects of Out of Sequence OFFs or NA Inputs to a Binary Encoder Object.

Input[1]	Input[2]	Input[3]	Input[4]	Input[5]	Input[6]	Output[3] MSB	Output[2]	Output[1] LSB
ON	OFF	ON	ON	ON	OFF	OFF	OFF	ON
ON	ON	ON	NA	ON	OFF	OFF	ON	ON
OFF	ON	ON	ON	ON	OFF	OFF	OFF	OFF
ON	ON	NA	ON	ON	OFF	OFF	ON	OFF
ON	ON	ON	ON	OFF	ON	ON	OFF	OFF

Example Application

The Binary Encoder object in this example is fed by a Sequence (6) object. The Sequence object receives an analog Input signal of 0 to 100% from a Loop object (not shown), and sequences ON its digital outputs in proportion to this heat demand. The Binary Encoder object converts the sequenced digital output values received from the Sequence (6) object into binary encoded combinations at Outputs[1] to [3], which control ON and OFF the three electric heat loads. The three physical loads driven in this example are electric heating coils sized at 500W, 1kW, and 2kW.



Figure–5.35 Example Binary Encoder Object Used in an Electric Heating Application.

In this example, using binary encoded digital logic with the three electric loads allows for six discrete levels of total heat output, as shown below Figure-5.52.

Table–5.52 Example Binary Encoder Object Inputs to Outputs in a Heating Application Example.

Heating Demand	Linear Sequenced Inputs						Binary Encoded Outputs			Total Heat (kW)
	Input[1]	Input[2]	Input[3]	Input[4]	Input[5]	Input[6]	Output[3] 2kW	Output[2] 1kW	Output[1] 500W	
0.0%	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	0.0kW
16.6%	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	0.5kW
33.3%	ON	ON	OFF	OFF	OFF	OFF	OFF	ON	OFF	1.0kW
50.0%	ON	ON	ON	OFF	OFF	OFF	OFF	ON	ON	1.5kW
66.7%	ON	ON	ON	ON	OFF	OFF	ON	OFF	OFF	2.0kW
83.3%	ON	ON	ON	ON	ON	OFF	ON	OFF	ON	2.5kW
100.0%	ON	ON	ON	ON	ON	ON	ON	ON	OFF	3.0kW

Refer to the Sequence (6) object for detailed information on its operation.

Binary Input

Object Usage: The Binary Input object provides a means for interfacing the application to any physical input point on a controller that allows a digital input signal. Typical use is to monitor the binary status of contact closures from various field devices, such as differential pressure sensors, flow switches, low temperature stats, or any other dry contact device. Each object includes a resettable counter that totals the number of status changes. If supported by the controller input, the Binary Input object also allows monitoring of a pulse train device such as a flow meter or demand meter. The Binary Input object monitors the assigned hardware input and determines the proper object output based upon the Binary Type selected. Binary Type selections are:

- Direct (Normally Open Contact)
- Reverse (Normally Closed Contact)
- Pulse (Pulse Train Device)

Device Support: (all firmware revisions)

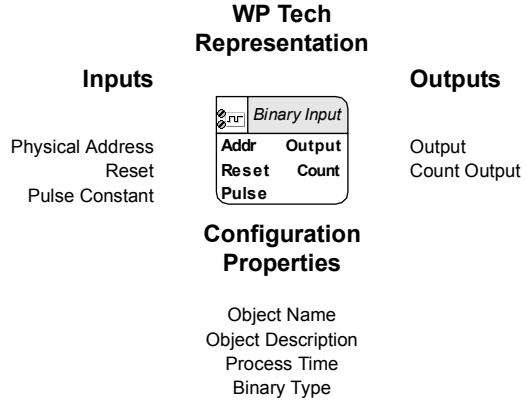
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 12 bytes
RAM: 22 bytes (standard controllers)
10 bytes (MN 800)

WP Tech Representation



Properties

Table–5.53 Binary Input Object Configuration Properties.

Abbrev.	Name	Class / Description	Def.	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.

Table–5.53 Binary Input Object Configuration Properties.

Abbrev.	Name	Class / Description	Def.	Range / Selections	Notes
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
Type	Binary Type	Class: Analog - This value defines the contact or device type connected to the hardware.	16	16 - Direct (N.O.) 17 - Reverse (N.C.) 18 - Pulse	NA or values outside valid range defaults to 16 (Direct).

Table–5.54 Binary Input Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Addr	Physical Address	Class: Analog - Indicates the physical hardware address (input terminal point on the controller) assigned to the Binary Input object. Either a DI (Digital Input) or UI (Universal Input) can be used.	Dependent on the controller platform selected.	If no physical hardware address is assigned (NA), the outputs are set to NA.
Reset	Reset	Class: Digital - Applies to all Binary Types: • For Direct and Reverse types, an ON causes the Count Output to be reset and held at zero, while an OFF allows the Count Output to operate. • For Pulse types, an ON causes both Output and the Count Output to be reset and held at zero, while an OFF allows both the Output and Count Output to operate.	—	A not active (NA) operates the same as OFF.
Pulse	Pulse Constant	Class: Analog - The user-supplied pulse scaling value used by the pulse algorithm when calculating a pulses per second frequency Output value. This input is active only when Binary Type is set to Pulse.	0.1 to 1000	A negative value or not active (NA) causes the Output and Count Output to be held at NA.

Table–5.55 Binary Input Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital / Analog - Digital if Type = Direct, Reverse; Analog if Pulse. • For Direct and Reverse types this reflects the current state of the hardware input, where: • Direct type: OFF at contact open, ON at contact close. • Reverse type: ON at contact open, OFF at contact close. • For Pulse types this is an analog value (rate) calculated by the pulse rate function, where Output = Pulse Constant x Pulses per Second.	Direct or Reverse: OFF (0) ON (100) Pulse: 0 to 16383

Table–5.55 Binary Input Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Count	Count Output	Class: Analog - Indicates the total status changes or pulses seen at the hardware input. The Count Output increments from 0 to 9,999 maximum, whereafter a count rollover occurs, resetting the Count Output to zero. The Count Output is also reset and held at zero by setting the Reset input to ON. This output indicates not active (NA) whenever the Binary Input object is not assigned a valid physical address, or the data from the assigned hardware point is not valid. NA also occurs if the Binary Type is pulse and the Pulse Constant is negative or NA. ***Note**:* The Count Output increments *only on a "contact open" that follows a "contact close."* If the contact is closed when the application is downloaded to the controller, this closed condition is not recognized by the counter after connecting online. Thus, on the first "contact open," the counter does not increment. However, once the contact closes again, the counter will resume incrementing, on the following "contact open."	0 to 9,999

Applying the Object

The Binary Input object monitors a two-state "dry contact-type" signal received on an I/A Series MicroNet controller digital or universal input (DI or UI). Examples of typical field hardware include differential pressure switches, flow switches, low temperature thermostats, contactor auxiliary contacts, as well as other dry contact devices. A pulse-meter device, such as a flow meter or demand meter, can also be monitored using a Binary Input object and a DI input on the controller. A pulse-meter generates a contact closure for each predefined quantity of delivered material, such as gallons or electrical power. Based on the time between input pulses, the Binary Input object outputs the current analog rate of delivery, such as gallons per hour or kilowatts. The Binary Input object also totals the number of pulses received on a resettable counter.

The object's Binary Type assignment configures the proper output condition based on these types:

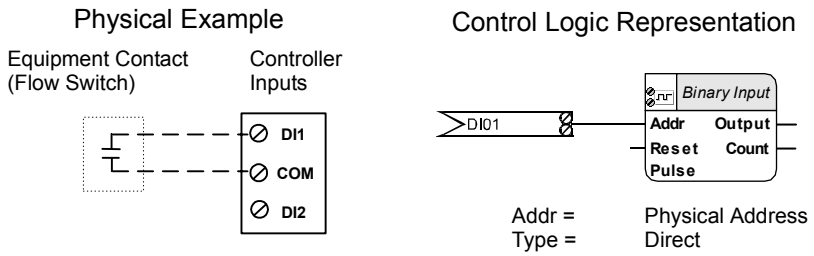
- Direct (Normally Open, or N.O. contact)
- Reverse (Normally Closed, or N.C. contact)
- Pulse Input Device, such as a pulse-based flow meter

Direct

A Binary Type selection of Direct causes the Output value to directly reflect the digital condition of the physical hardware input. An Inactive state is considered OFF while an Active state is considered ON. The following table Figure-5.56 and example object Figure-5.36 illustrate this.

Table–5.56 Direct Binary Type Binary Input Object Truth Table.

Physical Hardware	Physical State	Output
Invalid	—	Not Active (NA)
Open Contact	Inactive	Digital OFF
Closed Contact	Active	Digital ON



Figure–5.36 Example Binary Input Object for a Normally Open Contact Device.

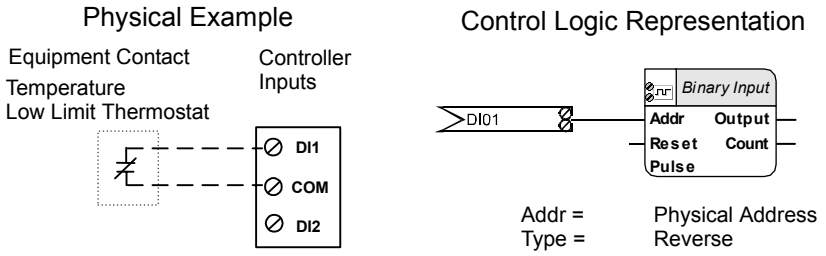
The example Binary Input object above is configured as Direct acting. The object monitors the N.O. flow switch contact and reports a switch closure (physical active state) as a digital ON at the Output, indicating proof of flow. The object also has an available Count Output, which reports the total number of status changes. This counter can be reset to zero by applying a digital ON to the Reset input of the object.

Reverse

A Binary Type selection of Reverse causes the Output value to inversely reflect the digital condition of the physical hardware input. An Inactive state is considered ON, while an Active state is considered OFF. The following table Figure-5.57 and example object Figure-5.37 illustrate this.

Table–5.57 Reverse Binary Type Binary Input Object Truth Table.

Physical Hardware	Physical State	Output
Invalid	—	Not Active (NA)
Open Contact	Active	Digital ON
Closed Contact	Inactive	Digital OFF



Figure–5.37 Example Binary Input Object for a Normally Closed Contact Device.

The example Binary Input object above is configured as Reverse acting. The object monitors the N.C. low limit thermostat contact and reports a contact open (physical inactive state) as a digital ON, indicating low limit temperature conditions. The object also has an available Count Output, which reports the total number of status changes. This counter can be reset to zero by applying a digital ON to the Reset input of the object.

Pulse

A Binary Type selection of Pulse is used to monitor a pulse-meter type device.

Note: The maximum pulse rate supported by an I/A Series MicroNet controller's digital input (DI) is 4 pulses per second. The minimum required pulse rate for standard controllers is one pulse per minute. Refer to Table–5.58 for details.

Table–5.58 Pulse Rates for I/A Series MicroNet Controllers.

Controller	Input	Pulse Rates[a]	
		Maximum	Minimum
MN 50, MN 100, MN 150, MN 200, and MN VAV	DI (All)	4 pulses/sec	1 pulse/1 min
	UI (All)	1 pulse/sec	1 pulse/1 min
MN 800	UI1 (Only)	10 pulses/sec	1 pulse/4 min
	UI (All Others)	1 pulse/sec	1 pulse/4 min

a. Pulse rates are based on a 50% duty cycle.

A pulse-meter device generates a contact closure for each predefined quantity of delivered material, such as gallons (liters) or electrical energy. This predefined quantity determines the Pulse Constant to the Binary Input object. Based on the value of the assigned Pulse Constant and the pulse rate of Input pulses, the Binary Input object calculates an Output analog value to represent the rate of delivery, where:

Output rate (per second) = Pulse Constant x Pulses per Second

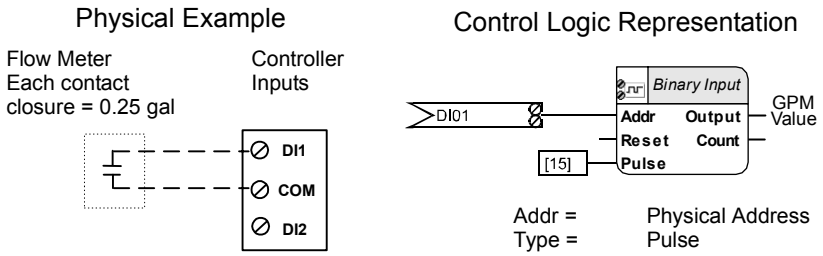
In cases where rate measurement is understood best in terms of minutes or hours, such as in gallons (or liters) per minute, or in kilowatts, the Pulse Constant can be multiplied by either 60 (minute) or 3600 (hour).

The maximum Pulse Constant allowed for a Binary Input object is 1000.0.

Output rate (per minute) = (Pulse Constant x 60) x Pulses per Second

Output rate (per hour) = (Pulse Constant x 3600) x Pulses per Second

For example, a flow meter is installed in a fuel line to measure the rate of fuel delivery. The meter produces a contact closure (pulse) for each 0.25 gallon (0.946 liter) of delivered fuel. A Binary Input object in a MicroNet controller monitors the physical contact in the flow meter.



Figure–5.38 Example Binary Input Object for a Pulse-Output Flow Meter.

If the Pulse Constant is given a value of 0.25, the Output of the object indicates a value representing gallons/second. In this application, the most useful flow rate is in gallons/minute (GPM), so the Pulse Constant is given a value of 15 (or 0.25 x 60) for the Output to indicate gallons/minute. If gallons/hour were required, a Pulse Constant of 900 (or 0.25 x 3600) could be used. The following chart shows some example outputs at some possible pulse rate input levels for each of the three Pulse Constants.

Table–5.59 Example Outputs at Pulse Rates Using Different Pulse Constants.

Input Pulse Rate Pulses/Sec.	Pulse Constant 0.25 (gal/sec)	Pulse Constant 15.0 (gal/min)	Pulse Constant 900 (gal/hr)
0.05 (every 20 sec.)	0.0125	0.75	4.5
0.1 (every 10 sec.)	0.025	1.5	90.0
0.5 (every 2 sec.)	0.125	7.5	450.0
1.0 (every sec.)	0.5	15.0	900.0
4.0 (4 times a sec.)	1.0	60.0	3600.0

In real applications, input pulse rates constantly fluctuate as the measured rate changes. The Binary Input pulse algorithm continuously monitors the Input and adjusts the Output value on each received pulse.

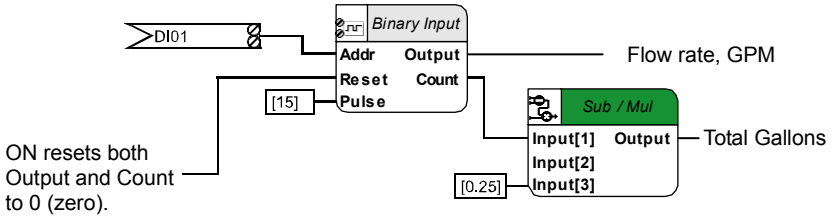
$$\text{Output} = \text{Pulse Constant} \times \frac{1}{(\text{time between last two pulses})}$$

This means the Output value will go higher if the time between pulses becomes shorter, or will go lower (decay) if the time between pulses is longer. Quicker pulse rates decay at a faster rate than slow pulse rates.

The Binary Input object uses an internal exponential low pass filter which limits the response of the output in relationship to a rapid pulse rate change at the Input. This filter dampens the effects of input change to output change to stabilize a noisy or rapidly changing input pulse signal rate.

Pulse Constant

In any Binary Input object, the total number of pulses is accumulated in the same way, regardless of the assigned Pulse Constant. The value of the counted pulses can increment from 0 to 9,999 and is available at the object's Count Output. In the previous flow meter example, each pulse represents 0.25 gallons. By using the Count Output as an Input to a Sub / Mul object math object, a running total of gallons is accumulated.



Figure–5.39 Example Count Output to Math Object for Material Totals.

This running total can be cleared by an ON to the Reset input on the Binary Input object. In this application example, this reset of total may occur at some periodic interval, such as daily, weekly, or monthly. If no reset signal is sent, the counter automatically would rollover from 9,999 back to zero and continue incrementing.

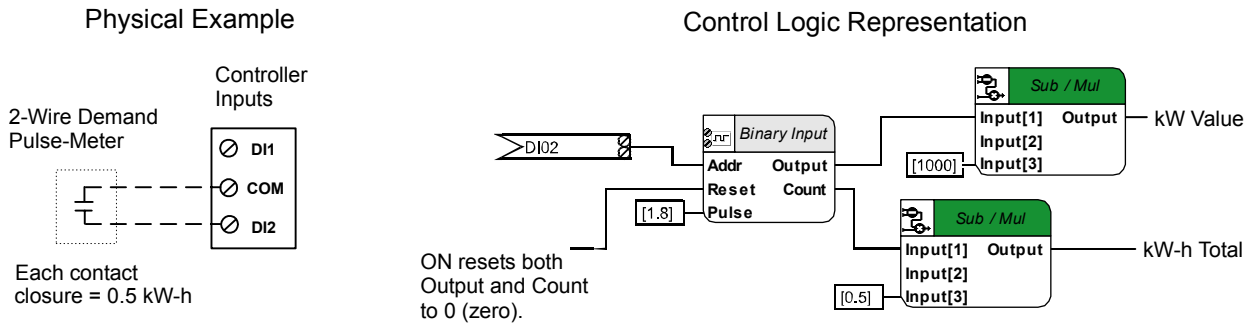
Pulse Demand Meter

Another typical Pulse application for a Binary Input object is to monitor an electrical demand pulse-meter. This type of meter is often supplied by the power utility to read the current power rate (kilowatts or kW) and the total energy accumulated (kilowatt-hours or kW-h). Both two-wire and three-wire demand meter types are used, with two-wire types most common.

- Two-wire pulse demand meters typically have a single N.O. contact.
- Three-wire pulse demand meters typically have two N.O. contacts and a common terminal. It is necessary only to monitor one contact.

The amount of kW-h per pulse is provided by the utility. With a two-wire pulse-meter, this kW-h value is the direct basis for the Pulse Constant. With a three-wire pulse-meter, this kW-h value must be doubled before using it as the basis for the Pulse Constant.

Demand Meter Example: This example is for a two-wire demand meter that delivers a contact closure (pulse) every 0.5 kW-h. In this case, the Binary Input object in the MicroNet Controller cannot directly report a kW output, because the necessary Pulse Constant (0.5 kW-h x (3600 sec/hr)), or 1800, would exceed 1000. However, MegaWatts (MW) can be indicated by dividing 1800 by 1000 for the Pulse Constant, and using the resulting 1.8 value.



Figure–5.40 Example Binary Input Object and Related Math Objects for a Pulse type Demand Meter.

The Binary Input object in this example Figure-5.40 uses two Sub / Mul math objects. The topmost object multiplies the MW output of the Binary Input object by the constant 1000 to produce a kW output value. The bottom object stores total energy usage (kW-h) by multiplying the number of pulses received from the Count Output with the 0.5 kW-h per pulse constant.

Binary Output

Object Usage: The Binary Output object provides a means for interfacing the application to a physical digital output point (DO) on a controller. Typical use is for start/stop control of equipment such as single speed fans, pumps, lights, or any controlled load requiring simple OFF / ON control. The Binary Output object monitors the single assigned digital input and determines the proper hardware output action based upon the Binary Action selected. Binary Action selections are:

- Direct (Normally Open Contact)
- Reverse (Normally Closed Contact)

Device Support: (all firmware revisions)

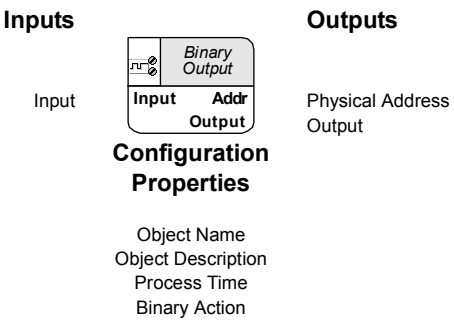
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.60 Binary Output Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.60 Binary Output Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Action	Binary Action	Class: Analog - This value defines the input to output action, either direct acting or reverse acting.	0	0 - Direct Acting 1 - Reverse Acting	MicroNet controller Digital Outputs (DOs) are Form-A only, which are N.O. when controller power is removed.

Table–5.61 Binary Output Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input	Input	Class: Digital - The Input value monitored to determine the value at the physical hardware and object Output. An unconnected or not active (NA) is evaluated as an input value of digital OFF, and the output value will be set to NA.	—	Typically a digital value, that is, OFF (0.0) or ON (100.0). Analog values are evaluated as: Negative (<0) = OFF Positive (>0) = ON

Table–5.62 Binary Output Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Addr	Physical Address	Class: Analog - Defines the physical hardware address (output terminal point on the controller) assigned to the Binary Output object.	Dependent on the controller platform selected.
Output	Output	Class: Digital - This output indicates the calculated digital output state.	OFF (0.0) ON (100.0) NA Unconnected or NA

Applying the Object

The Binary Output object is used to operate a physical two-state output point on an I/A Series MicroNet controller known as a digital output (DO). Using a Binary Output object, control is OFF/ON, with typical controlled devices including single speed fans, pumps, lights, or any two-state load device.

Action

The Binary Output object monitors the assigned digital Input value and determines the proper hardware (DO) output action based on the selected Binary Action, which is either:

- Direct
- Reverse

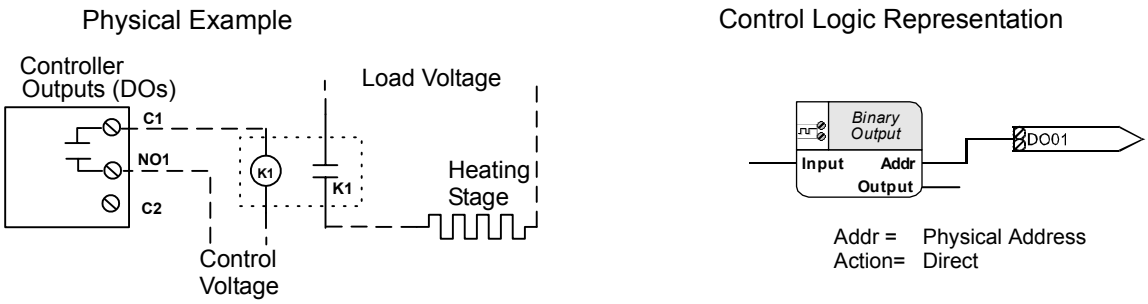
Each binary action is explained ahead.

Direct

A Binary Action selection of Direct causes the DO and Output value to directly follow the digital Input value. An unconnected or not active (NA) Input is considered a digital OFF. The following table Figure-5.63 and example object Figure-5.41 illustrate.

Table–5.63 Direct Binary Output Object Truth Table.

Input	Controller DO	Output
OFF	Open	Digital OFF
ON	Closed	Digital ON
Unconnected or NA	Open	NA
Controller Power OFF	Open	—



Figure–5.41 Example Direct Acting Binary Output Object (Electrical Heat Load).

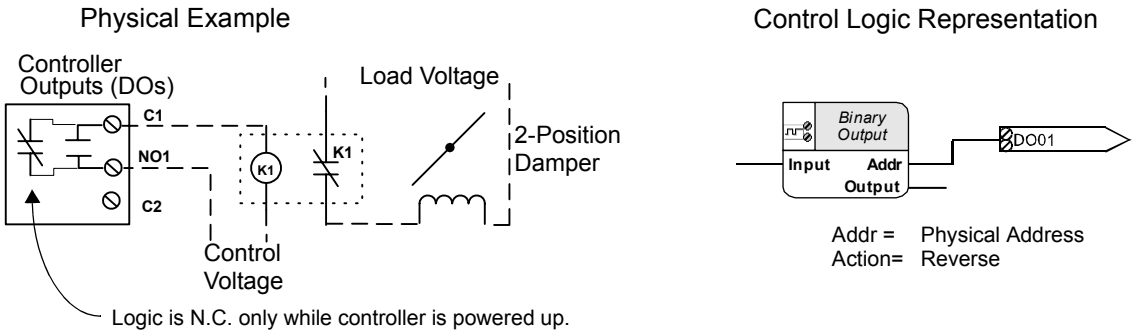
The example Binary Output object above is configured as Direct acting. The object follows the digital logic on the Input and switches the physical hardware output (DO) and digital Output to match.

Reverse

A Binary Type selection of Reverse causes the DO and Output value to inversely follow the digital Input value. An unconnected or not active (NA) Input is considered a digital OFF. The following table Figure-5.64 and example object Figure-5.42 illustrate this.

Table–5.64 Reverse Binary Output Object Truth Table.

Input	Controller DO	Output
OFF	Closed	Digital ON
ON	Open	Digital OFF
Unconnected or NA	Closed	NA
Controller Power OFF	Open	—



Figure–5.42 Example Reverse Acting Binary Output Object (Two-position Damper Actuator).

The example Binary Output object above is configured as Reverse acting. The object reads the digital logic on the Input and switches the physical hardware output (DO) and digital Output to the reverse condition.

Note: DO contacts assigned to Reverse acting Binary Output objects are normally closed (N.C.) only while the controller is powered up.
If power to the controller is lost, all controller DO contacts are open.

Calendar

Object Usage: The Calendar object provides a means to program annual exception-day events such as holidays and periods requiring a change from normal schedule operation. Repeating holidays can be defined as either calendar-based (specific dates) or occurrence-based (set by day of week). Each Calendar object supports up to 8 calendar events or occurrences (Programs 1 to 8).

Programs can be assigned to one or more of the four digital (OFF/ON) outputs. This allows a particular output to follow a predefined sequence that reflects the results of multiple events or occurrences. Programs are defined by the various configuration properties, such as start and end dates, times, durations, assigned outputs, etc.

The “Force Exception” inputs allow direct override control of the associated outputs. An input of ON or OFF is directly reflected at the associated output, regardless of calculated conditions. A not active (NA) input allows the associated output to be under control of the normal Calendar object calculation.

The Calendar Enable input allows normal operation of the object whenever its input value is NA or ON. While the Calendar Enable input value is OFF, all outputs remain OFF and other inputs are ignored.

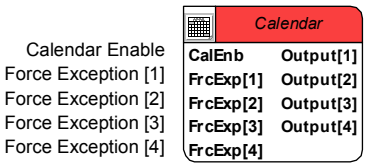
Device Support:

MN 800 series

Memory Requirements: (per object)

EEPROM: 142 bytes
RAM: 8 bytes

WP Tech Representation



Configuration Properties

Object Name
Object Description
Program Type [1]
to
Program Type [8]
Start Date [1]
to
Start Date [8]
Start Time [1]
to
Start Time [8]
End Date [1]
to
End Date [8]
End Time [1]
to
End Time [8]
Day Select [1]
to
Day Select [8]
Duration [1]
to
Duration [8]
Output Select[1]
to
Output Select [8]

WP Tech Stencil:
Schedule Control

Properties

Table–5.65 Calendar Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.

Table–5.65 Calendar Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
PgmTyp[1]	Program Type [1]	Class: Analog - Defines Program 1 operation (Event or Occurrence).	0	For each: Event (0) or Occurrences : 1st (1) 2nd (2) 3rd (3) 4th (4) 5th (5) Last (6) All (7)	Event = Start and stop by defined dates and times. Occurrence = Start and stop by a combination of factors including day of week, occurrence number, duration, and time.
PgmTyp[2]	Program Type [2]	Class: Analog - Defines Program 2 operation (Event or Occurrence).	0		
⋮	⋮	⋮	⋮		
PgmTyp[8]	Program Type [8]	Class: Analog - Defines Program 8 operation (Event or Occurrence).	0		
StrtDate[1]	Start Date [1]	Class: Analog - Defines for Program 1: Event — start date Occurrence — starting month	NA	For each: MM / DD format (month / day), where, MM = 01 to 12 DD = 01 to 31 (01/01 to 12/31)	Start Date is considered invalid if set to not active (NA) or a date out of calendar range (for example, 02/31). An invalid Start Date sets the associated program to remain inactive. The associated output(s) are not influenced by the program.
StrtDate[2]	Start Date [2]	Class: Analog - Defines for Program 2: Event — start date Occurrence — starting month	NA		
⋮	⋮	⋮	⋮		
StrtDate[8]	Start Date [8]	Class: Analog - Defines for Program 8: Event — start date Occurrence — starting month	NA		
StrtTm[1]	Start Time [1]	Class: Analog - Defines for Program 1: Event — start time Occurrence — start time	NA	For each: HH: MM (hours/ mins), where, HH = 00 to 23 MM = 00 to 59 (00:00 to 23:59)	Start Time is considered invalid if set to not active (NA) or a time out of 24-hour clock range. An invalid Start Time sets the associated program to remain inactive. The associated output(s) are not influenced by the program.
StrtTm[2]	Start Time [2]	Class: Analog - Defines for Program 2: Event — start time Occurrence — start time	NA		
⋮	⋮	⋮	⋮		
StrtTm[8]	Start Time [8]	Class: Analog - Defines for Program 8: Event — start time Occurrence — start time	NA		
EndDate[1]	End Date [1]	Class: Analog - Defines for Program 1: Event — end date Occurrence — end month	NA	For each: MM / DD format (month / day), where, MM = 01 to 12 DD = 01 to 31 (01/01 to 12/31)	End Date is considered invalid if set to not active (NA) or a date out of calendar range (for example, 02/31). An invalid End Date sets the associated program to remain inactive. The associated output(s) are not influenced by the program.
EndDate[2]	End Date [2]	Class: Analog - Defines for Program 2: Event — end date Occurrence — end month	NA		
⋮	⋮	⋮	⋮		
EndDate[8]	End Date [8]	Class: Analog - Defines for Program 8: Event — end date Occurrence — end month	NA		

Table–5.65 Calendar Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
EndTm[1]	End Time [1]	Class: Analog - Defines for Program 1: Event — end time Occurrence — not used	NA	For each: HH: MM (hours/ mins), where, HH = 00 to 23 MM = 00 to 59 (00:00 to 23:59)	Not used if program is Occurrence-based. End Time is considered invalid if set to not active (NA) or a time out of 24-hour clock range. An invalid End Time sets the associated program to remain inactive.The associated output(s) are not influenced by the program.
EndTm[2]	End Time [2]	Class: Analog - Defines for Program 2: Event — end time Occurrence — not used	NA		
⋮	⋮	⋮	⋮		
EndTm[8]	End Time [8]	Class: Analog - Defines for Program 8: Event — end time Occurrence — not used	NA		
DaySel[1]	Day Select [1]	Class: Analog - Defines for Program 1: Event — not used Occurrence — day of week on which occurrence begins	NA	For each: Sun. (0) Mon. (1) Tue. (2) Wed. (3) Thu. (4) Fri. (5) Sat. (6)	Not used if program is Event-based. A not active (NA) or out-of-range Day Select causes the programmed occurrence to remain inactive.The associated output(s) are not influenced by the program.
DaySel[2]	Day Select [2]	Class: Analog - Defines for Program 2: Event — not used Occurrence — day of week on which occurrence begins	NA		
⋮	⋮	⋮	⋮		
DaySel[8]	Day Select [8]	Class: Analog - Defines for Program 8: Event — not used Occurrence — day of week on which occurrence begins	NA		
Duration[1]	Duration [1]	Class: Analog - Defines for Program 1: Event — not used Occurrence — length of occurrence (in hours)	NA	For each: 0.0 to 1000 (hours)	Not used if program is Event-based. A not active (NA) duration causes the programmed occurrence to remain inactive.The associated output(s) are not influenced by the program.
Duration[2]	Duration [2]	Class: Analog - Defines for Program 2: Event — not used Occurrence — length of occurrence (in hours)	NA		
⋮	⋮	⋮	⋮		
Duration[8]	Duration [8]	Class: Analog - Defines for Program 8: Event — not used Occurrence — length of occurrence (in hours)	NA		

Table–5.65 Calendar Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
OutSel[1]	Output Select [1]	Class: Analog - Defines the output(s) assigned to Program 1.	NA	For each: Any combination: Output [1] Output [2] Output [3] Output [4]	A not active (NA) or out-of-range Output Select value causes the associated programmed event or occurrence to remain inactive.
OutSel[2]	Output Select [2]	Class: Analog - Defines the output(s) assigned to Program 2.	NA		
⋮		⋮	⋮		
OutSel[8]	Output Select [8]	Class: Analog - Defines the output(s) assigned to Program 8.	NA		

Table–5.66 Calendar Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
CalEnb	Calendar Enable	Class: Digital - A not active (NA) or ON is required for normal Calendar object operation. If this input is OFF, all Outputs [1] to [4] are held OFF, and Force Exception inputs are ignored.	—	
FrcExp[1]	Force Exception [1]	Class: Digital - A not active (NA) is required for Output [1] to follow normal calendar control. Otherwise Output [1] directly follows the digital state at this input, regardless of calendar conditions.	—	If this input is OFF, Output [1] is held OFF. If this input is ON, Output [1] is held ON.
FrcExp[2]	Force Exception [2]	Class: Digital - A not active (NA) is required for Output [2] to follow normal calendar control. Otherwise Output [2] directly follows the digital state at this input, regardless of calendar conditions.	—	If this input is OFF, Output [2] is held OFF. If this input is ON, Output [2] is held ON.
FrcExp[3]	Force Exception [3]	Class: Digital - A not active (NA) is required for Output [3] to follow normal calendar control. Otherwise Output [3] directly follows the digital state at this input, regardless of calendar conditions.	—	If this input is OFF, Output [3] is held OFF. If this input is ON, Output [3] is held ON.
FrcExp[4]	Force Exception [4]	Class: Digital - A not active (NA) is required for Output [4] to follow normal calendar control. Otherwise Output [4] directly follows the digital state at this input, regardless of calendar conditions.	—	If this input is OFF, Output [4] is held OFF. If this input is ON, Output [4] is held ON.

Table–5.67 Calendar Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital - Reflects the calculated OFF or ON state of the Calendar object for Output [1]. This output state is based upon the associated input force condition and the results of the current calendar events/occurrences defined to use this output.	OFF (0.0) ON (100.0)
Output[2]	Output [2]	Class: Digital - Reflects the calculated OFF or ON state of the Calendar object for Output [2]. This output state is based upon the associated input force condition and the results of the current calendar events/occurrences defined to use this output.	OFF (0.0) ON (100.0)
Output[3]	Output [3]	Class: Digital - Reflects the calculated OFF or ON state of the Calendar object for Output [3]. This output state is based upon the associated input force condition and the results of the current calendar events/occurrences defined to use this output.	OFF (0.0) ON (100.0)

Table–5.67 Calendar Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[4]	Output [4]	Class: Digital - Reflects the calculated OFF or ON state of the Calendar object for Output [4]. This output state is based upon the associated input force condition and the results of the current calendar events/occurrences defined to use this output.	OFF (0.0) ON (100.0)

Applying the Object

The Calendar object provides a means to program annual exception-day events such as holidays and periods requiring a change from normal schedule operation. Repeating holidays can be defined as either event-based or occurrence-based. For example, an event-based holiday might be the Fourth of July. An example of an occurrence-based holiday might be the first Monday of a particular month. Each Calendar object supports up to 8 calendar events or occurrences (*programs* 1 to 8).

Program Type

Events or occurrences (programs) 1 through 8 must each be defined as event-based or occurrence-based, using the Program Type [1] through [8] configuration property. Use of all eight programs is not required for correct operation.

Event-Based

Assigning a value of 0 (Event) as the Program Type causes the program to perform a calendar *event*. A programmed calendar event utilizes the program's Start Date / Start Time and End Date / End Time configuration parameters to create a calendar period during which the event will be active. A programmed event results in a Digital ON condition whenever the actual device's clock / calendar is found to be within the defined calendar period. The active outputs are assigned through the use of the Output Select parameter.

To be valid, a programmed event must have a valid Start Date / Start Time and End Date / End Time. An invalid Start Date or End Date, or any date which is out of calendar range, causes the associated program to remain inactive. An inactive program will not influence any assigned output(s). An invalid Start Time or End Time, or any time which is out of the 24 hour clock range, is likewise defined as inactive.

The Calendar object is designed to handle events whose active periods span the end of one calendar year and the beginning of the next. This is the case when the Start Date / Start Time is set later than the End Date / End Time, for example an event which has a start date of October 3 and an end date of February 14. This feature eliminates the need to create multiple programs for an event that spans the first of the calendar year.

An event with an identical Start Date / Start Time and End Date / End Time will cause that specific calendar event to remain inactive. The associated outputs are not influenced by that programmed event.

Occurrence-based

Assigning a value of 1 (1st), 2 (2nd), 3 (3rd), 4 (4th), 5 (5th), 6 (Last), or 7 (All) as the Program Type causes the program to perform a calendar *occurrence*. A programmed calendar occurrence utilizes the program's Start Date / Start Time, End Date, Day Select, and Duration configuration parameters to create a calendar period during which the occurrence is active. The Calendar object uses the months assigned to the program's Start Date and End Date to validate the occurrence period. The Day Select and Duration configuration parameters are used to set up the period during which the occurrence will be active. A programmed occurrence results in a Digital ON condition whenever the actual device's clock / calendar is found to be within the calculated period. The above combination establishes the repetitiveness of the occurrence within the programmed Start Date and End Date validation period. The active outputs are assigned through the use of the Output Select parameter.

The Start Date configuration parameter defines the start of the active validation period and is expressed in month and date format. The Start Time configuration parameter defines the hour and minute at which the program occurrence begins. The End Date configuration parameter defines the end of the active validation period, and is expressed in month and date format. In calculating the active occurrence, the algorithm uses only the month portion of the month / date assigned to the Start Date and End Date.

To be valid, a programmed occurrence must have a valid Start Date / Start Time and End Date. An invalid Start Date, or any date which is out of calendar range, causes the associated program to remain inactive. An inactive program will not influence any assigned output(s). An invalid Start Time or End Time, or any time which is out of 24 hour clock range, is likewise defined as inactive.

The Program Type configuration parameter is set to reflect the required occurrence within a particular month or months. Within a given month, this parameter can be set to indicate a single occurrence (the 1st, 2nd, 3rd, 4th, 5th, or last) or all occurrences. A programmed occurrence that specifies an active period that may not occur within the month (for example, the 5th occurrence in a month that contains only four) causes the algorithm to ignore the occurrence for that month.

The Day Select configuration parameter is set to define the day of the week (Sunday, Monday, Tuesday, etc.) on which the occurrence is set to begin. A not active (NA) or out of range value causes the programmed occurrence to remain inactive. An inactive program will not influence any assigned output(s).

The Duration configuration parameter defines the length of the occurrence, in hours, with a range of 0.0 to 1000.0 hours. Once an occurrence is activated, the algorithm uses the Duration value to determine its completion date and time. If the duration is sufficiently long, the occurrence is allowed to continue into the following month, even when it is outside the validation period (the month obtained from End Date). An assigned Duration that causes the occurrence to exceed or overlap the next scheduled occurrence will automatically use the most recent occurrence as the basis for active output calculation. Negative or not active (NA) Duration values cause the programmed occurrence to remain inactive. An inactive program will not influence any assigned output(s).

The Calendar object algorithm is designed to handle occurrences whose validation periods span the end of one calendar year and the beginning of the next. This is the case when the Start Date month is set later than the End Date month. This feature eliminates the need to create multiple programs to span from one calendar year to the next.

Event/Occurrence Required Properties

Table–5.68, below, shows the configuration properties required for proper operation of programmed calendar events and occurrences. Properties that are indicated as “Not Required” have no effect on the program’s operation. Invalid values including not active (NA) will cause the individual program to remain inactive where the associated output or outputs are not influenced by the program.

Table–5.68 Required Configuration Properties for Programmed Events and Occurrences.

Program Type	Configuration Properties					
	Start Date	Start Time	End Date	End Time	Day Select	Duration
Event	Required	Required	Required	Required	Not Required	Not Required
Occurrence	Required	Required	Required	Not Required	Required	Required

Leap Year Operation

Some leap year considerations that must be kept in mind are:

- If February 29 is directly involved with a calendar event, the user must program events to specifically use this particular date.
- An event that does not specifically use February 29 as a Start Date, End Date, or both will automatically include this date in the active event calculation whenever leap year conditions apply.
- An occurrence that uses February as the month in which it is active automatically uses February 29 for active occurrence calculations whenever leap year conditions apply.

An event that specifies February 29 as a Start Date, End Date, or both will be evaluated according to the following:

Event with Start Date of February 29

Setting the Start Date to February 29, and the End Date to a value that exceeds February 29, causes the algorithm to operate the event *only during leap years*.

Example: Start Date = February 29 and End Date = March 15.

Setting the Start Date to February 29, and the End Date to a value that precedes February 29, causes the algorithm to operate the event *starting in a leap year and concluding in the following year*.

Example: Start Date = February 29 and End Date = January 15.

Event with End Date of February 29

Setting the End Date to February 29, and the Start Date to a value preceding February 29, causes the algorithm to operate the event *only during leap years*.

Example: Start Date = January 15 and End Date = February 29.

Setting the End Date to February 29, and the Start Date to a value that exceeds February 29, causes the algorithm to operate the event *concluding in a leap year after initiating the event in the previous year*.

Example: Start Date = March 15 and End Date = February 29.

Event with both Start Date and End Date of February 29

Setting both the Start Date and End Date to February 29 causes the Calendar object algorithm to use the assigned Start Time and End Time to determine event operation.

Setting the Start Time to a value that precedes the End Time causes the algorithm to operate the event only during leap years.

Example: Start Time = 6:00 AM and End Time = 9:00 PM.

Setting the Start Time to a value that exceeds the End Time causes the algorithm to initiate the event in a leap year and conclude the event the following leap year.

Example: Start Time = 9:00 PM and End Time = 6:00 AM.

Examples

This example illustrates the setup required to execute the following events and occurrences:

- New Years Day
- Memorial Day
- Independence Day
- Labor Day
- Thanksgiving
- Christmas
- Christmas Break
- Meeting (Second Tuesday — September through May)

Events will be programmed for New Years Day, Independence Day, and Christmas. Occurrences will be programmed for Memorial Day, Labor Day, and Thanksgiving. These events and occurrences will all be set to activate the same output (Output 1), because each holiday requires the same schedule performance (unoccupied).

An additional event will be programmed to handle a Christmas break that is scheduled to span from one year to the next. The event will be set to activate Output 3.

A repetitive occurrence will be programmed to handle a meeting that is schedule for the second Tuesday of each month. The valid activation period will be programmed to span the months of September through May. The occurrence is set to activate Outputs 2 and 3.

For this example, the Calendar object configuration parameters are set as follows:

Events and Occurrences	Program Type	Start Date	Start Time	End Date	End Time	Day Select	Duration	Output Select
New Years Day	Event	01/01	00:00	01/01	23:59	—	—	1
Memorial Day	Last Occur	05/01	00:00	05/31	—	Monday	24.0	1
Independence Day	Event	07/04	00:00	07/04	23:59	—	—	1
Labor Day	1st Occur	09/01	00:00	09/30	—	Monday	24.0	1
Thanksgiving	4th Occur	11/01	00:00	11/31	—	Thursday	24.0	1
Christmas	Event	12/25	00:00	12/25	23:59	—	—	1
Christmas Break	Event	12/24	00:00	1/16	23:59	—	—	3
Meeting	2nd Occur	09/01	00:00	05/31	—	Tuesday	24.0	2 and 3

Clocked SR

Object Usage: The Clocked SR object is a three-input logic object for use with digital values (DV). It performs a clocked set-reset flip flop logic function. In operation, the Clocked SR object is similar to the two-input SR Flip-Flop object (page 504). However, the Clocked SR uses an additional Clock input that requires a coinciding OFF/ON transition (clock) before Output changes. An unconnected input is considered invalid or not active (NA), and is ignored in the object's algorithm. If the Set and Reset inputs are NA, the output is set to NA.

Output = Clocked Set-Reset Flip Flop

Device Support: (all firmware revisions)

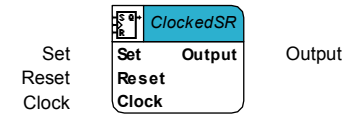
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 14 bytes (standard controllers)
4 bytes (MN 800)

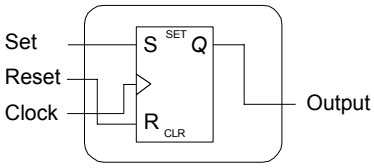
WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

Logic



Clocked SR

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Digital Logic Objects

Object Name	Digital Object Algorithm (all are three-input unless noted)
AND / AND	In1 AND In2 AND In3
AND / OR	(In1 AND In2) OR In3
Clocked SR	Clocked Set-Reset Flip-Flop Logic
EXOR	Two-input, Exclusive OR
Latch	Digital Sample and Hold or Latch
OR / AND	(In1 OR In2) AND In3
OR / OR	In1 OR In2 OR In3
SR Flip-Flop	Two-input, Set-Reset Flip-Flop Logic

Properties

Table–5.69 Clocked SR Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.70 Clocked SR Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Set	Set	Class: Digital - Evaluated as the Set input. A not active (NA) is disregarded by the Clocked SR algorithm.	—	See the Truth Table and Timing Diagram for input to output sequence of operation.
Reset	Reset	Class: Digital - Evaluated as the Reset input. A not active (NA) is disregarded by the Clocked SR algorithm.	—	
Clock	Clock	Class: Digital - Evaluated as the Clock input. Clock action occurs on detection of rising edge.	—	

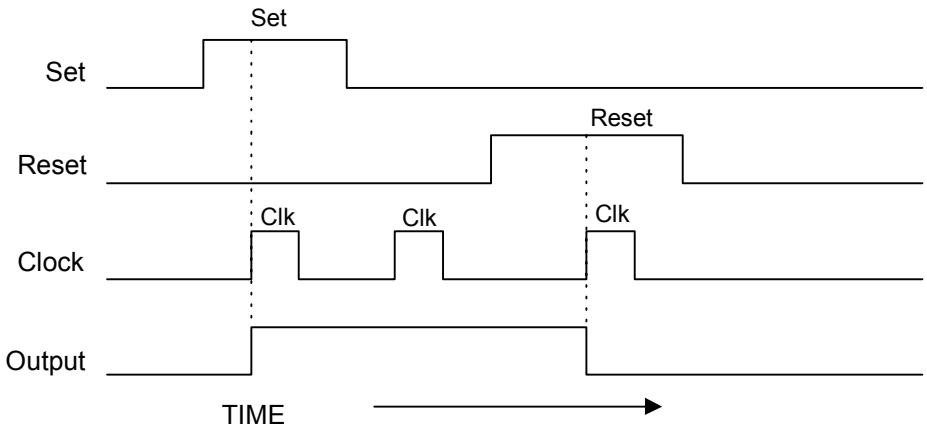
Table–5.71 Clocked SR Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The output indicates the result of the Clocked SR logic algorithm. If not active (NA) is present at the Set and Reset inputs, the output is set to NA.	OFF (0.0) ON (100.0)

Applying the Object

The Clocked SR object is a three-input logic object that processes digital values (DV for OFF or ON) and produces a DV output. It is most similar to the two-input SR Flip-Flop object. Both the Clocked SR object and SR Flip-Flop object toggle (flip-flop) their object Output with DV changes on the Set and Reset inputs. The Clocked SR object is a unique use of a third input, Clock, which requires an OFF/ON transition (clock) to coincide with a Set or Reset input change in order for the Output to be toggled.

The following timing diagram Figure-5.43 and truth table Figure-5.72 help illustrate the operation of the Clocked SR object.



Figure–5.43 Timing Diagram for a Clocked SR Object.

Table–5.72 Truth Table for Clocked SR Object.

Action	Inputs			Output
	Set	Reset	Clock	
Hold	OFF	OFF	Rising edge of clock	No change
Set	ON	OFF	Rising edge of clock	ON
Reset	OFF	ON	Rising edge of clock	OFF
Prohibited	ON	ON	Rising edge of clock	No change

Note: Following a controller reset the object Output is NA until an object input forces Set or Reset to ON or OFF.

As with other logic objects, Inputs to this object are typically digital values from a connection to another object's output, represented numerically with a 0 (OFF) or 100 (ON). However, Inputs can also process analog values or a value from an assigned constant. An Input with an analog value or constant is evaluated as OFF if the value is negative or equal to zero (0.0), or ON if the Input has any positive value greater than zero.

Compare

Object Usage: The Compare object compares the analog value on the Input to the analog value on each Compare input (A and B), and produces a digital ON output whenever any exact match is found. Otherwise, the digital output is OFF. If the Input is not active (NA), the output is set to NA.

Output = ON if Input value is exact match to Compare A value or Compare B value.

The compare function provided by this object is similar to one included in the Compare 2 object (page 192) (available for any Rev.3 or higher controller).

Device Support: (all firmware revisions)

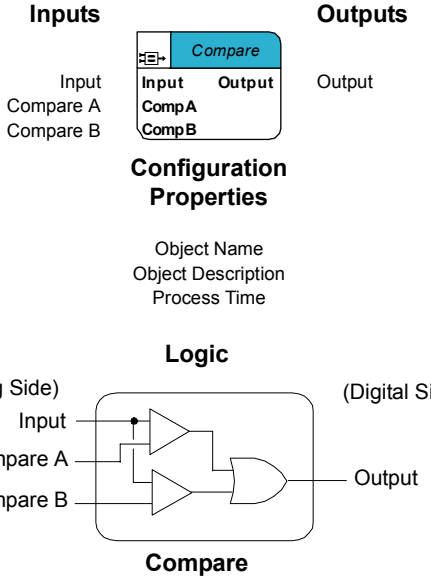
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Logic and Math Control

Properties

Table–5.73 Compare Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.74 Compare Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Analog - The input value that is compared against valid values at both inputs Compare A and Compare B.	-163.83 to 16383	A not active (NA) causes the output to be set to NA.
CompA	Compare A	Class: Analog - One of the two inputs whose value is compared to the value on the Input. An exact match sets the Output to ON.	-163.83 to 16383	A not active (NA) is disregarded by the compare algorithm.
CompB	Compare B	Class: Analog - One of the two inputs whose value is compared to the value on the Input. An exact match sets the Output to ON.	-163.83 to 16383	A not active (NA) is disregarded by the compare algorithm.

Table–5.75 Compare Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The output indicates the result of the Compare logic algorithm. The output is set to ON if the value at the Input is an exact match to the value at either input Compare A or Compare B. The output is set to not active (NA) whenever the input is NA.	OFF (0.0) ON (100.0)

Applying the Object

The Compare object is used to test an input value against two possible other values and issue a digital ON whenever an exact match is found. The input side of the object is analog and has three inputs; the Input to be tested and two Compare (A and B) inputs. The output side of the object is digital, with only a single OFF or ON Output.

Inputs to the Compare object are typically analog values, however digital values can be used as well; they are evaluated numerically as 0.0 (OFF) or 100.0 (ON). Table–5.76 demonstrates the Compare object operation.

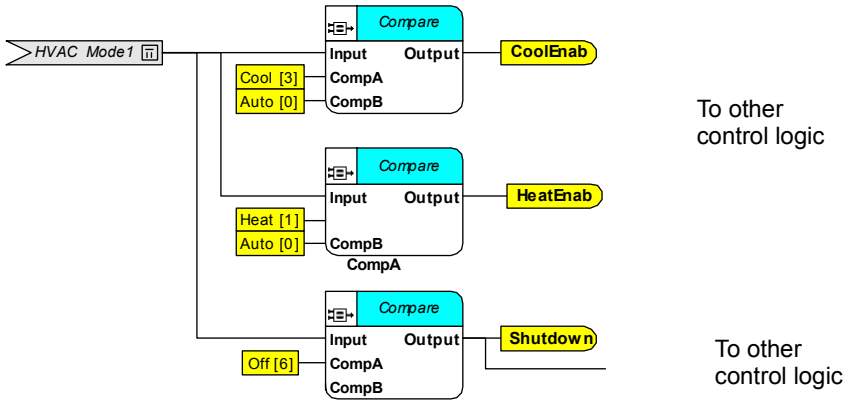
Table–5.76 Truth Table for Compare Object.

Analog Type Inputs			Digital Output
Input	Compare A	Compare B	
Valid value, but not equal to A or B	Valid value A	Valid value B	OFF
Valid value A	Valid value A	Ignored or NA	ON
Valid value B	Ignored or NA	Valid value B	ON
NA	Ignored or NA	Ignored or NA	NA

Example Application

A typical control logic application for the Compare object is to test an enumerated value, meaning an output that produces several discrete values to indicate different conditions. An example is the HVAC Mode tag for a MicroNet digital wall sensor, which produces one of these values in response to a sensor user's input: 0 (AUTO), 1 (HEAT), 3 (COOL), 6 (OFF).

Compare objects can isolate one or more of these enumerated values for a particular use in the control application, as shown below in Figure–5.44.



Figure–5.44 Example Compare Objects Used for Enumerated Value Testing.

In the example above, all three Compare objects evaluate the output of the same MicroNet sensor HVAC Mode tag against assigned constant values. The top two Compare objects both produce an ON if the sensor has selected the AUTO mode, otherwise, only one of the three Compare objects produces an ON as a result of a particular sensor mode override.

Compare 2

(Rev.3 or Later Firmware Required)

Object Usage: The Compare 2 object compares the values present at analog Inputs[1] and [2] in two different ways, and produces digital outputs at Outputs[1] and [2] using the following criteria:

Output[1]:

If Input[1] value > Input[2] value, Output[1] = ON.
If Input[1] value < Input[2] value, Output[1] = OFF.

Output[2]:

The Tolerance input is used in a “window-type” comparison between the values at Input[1] and [2]. If Input[1] value = Input[2] value ± Tolerance value, Output[2] = ON; otherwise, Output[2] = OFF.
If Tolerance input = 0, NA, or negative, Input[1] must equal Input[2] before Output[2] = ON.

Outputs[1] and [2] are initialized to OFF following a controller reset, or if one Input[1] or [2] is set to not active (NA). If both Inputs[1] and [2] have an NA, both Outputs[1] and [2] are set to NA.

Device Support: (See page 7)

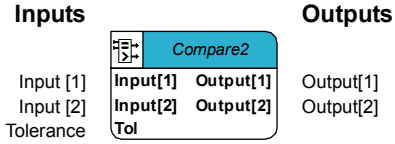
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F2, F3, H2, H3, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx
where xx = V2 or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 14 bytes (standard controllers)
4 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Properties

Table–5.77 Compare 2 Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.

Table–5.77 Compare 2 Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.78 Compare 2 Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The input value compared against the value at Input[2] for the object algorithm to set the digital outputs accordingly.	-163.83 to 16383	A not active (NA) sets both outputs to OFF. If Inputs[1] and [2] are both NA then Outputs[1] and [2] are both set to NA.
Input[2]	Input [2]	Class: Analog - The input value compared against the value at Input[1] for the object algorithm to set the digital outputs accordingly.	-163.83 to 16383	
Tol	Tolerance	Class: Analog - Used in the object algorithm to set the state of Output[2]. Only positive numbers are evaluated. The Tolerance value becomes a “plus or minus window” used in the comparison between the values at Inputs[1] and [2].	0 to 16383	A negative or not active (NA) value is evaluated the same as 0 (zero). (No Tolerance)

Table–5.79 Compare 2 Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital - Is set to ON whenever input values are as follows: • Input[1] > Input[2] Otherwise, this output is OFF whenever: • Input[1] < Input[2], or • Input[1] or [2] is not active (NA). If both Inputs[1] and [2] have an NA, this output is set to NA.	OFF (0.0) ON (100.0)
Output[2]	Output [2]	Class: Digital - Is set to ON whenever input values are as follows: • Input[1] > Input[2] - Tolerance AND Input[1] < Input[2] + Tolerance Otherwise, this output is OFF whenever input values are as follows: • Input[1] < Input[2] - Tolerance, or • Input[1] > Input[2] + Tolerance, or • Input[1] or [2] is not active (NA). If both Inputs[1] and [2] have an NA, this output is set to NA.	OFF (0.0) ON (100.0)

Applying the Object

The Compare 2 object provides two separate compare functions based on the values received on inputs. Each function has a dedicated digital output:

- Output[1] provides the results of a simple “greater than” function, whereby it is ON only while the value at Input[1] is greater than Input[2]. If the value at Input[1] is equal to or less than Input[2], Output[1] is OFF.
- Output[2] provides the results of a “plus-or-minus window” compare function, whereby the value at the Tolerance input is used. Output[2] is ON only if the value at Input[1] equals the Input[2] value, plus or minus the Tolerance value. If the plus-or-minus amount between Inputs[1] and [2] is greater than Tolerance, Output[2] is OFF.

Note: If the Tolerance input is left unconnected (not active or NA), the Output[2] function is identical to that provided by the Compare object, that is, Input[1] must be equal to Input[2] to produce an ON at Output[2].

Reset and Not Active

Upon reset, both outputs are initialized to OFF before the object executes. Whenever a not active (NA) is at Input[1] or [2], both outputs are set to OFF. If both Inputs[1] and [2] have an NA, both outputs are set to NA.

Control Override

Object Usage: The Control Override object provides a method for overriding a digital control signal for a programmable period from 0.02 to 240 minutes. A timed override results from a digital ON state transition at the Override Input. The timed override feature has Inputs for Override Enable, Override Cancel, and the Override Time period. An active override is indicated on the Time Remaining output as the remaining number of minutes in the override.

Device Support: (all firmware revisions)

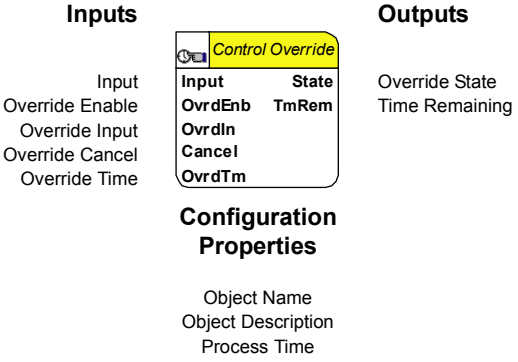
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 14 bytes
RAM: 20 bytes (standard controllers)
6 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.80 Control Override Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.81 Control Override Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input	Input	Class: Digital - The digital signal that is “or’d” with the timer override sequence. If unconnected or not active (NA), the remaining Inputs are used for a timed override.	—	
OvrdEnb	Override Enable	Class: Digital - Enables or disables the timed override feature. An ON allows a timed override to be initiated at the Override Input. An OFF disables all timed override Inputs and holds the Time Remaining output at zero.	—	If unconnected or NA, the Time Remaining output is NA and a timed override is not available. The Input (only) determines the Override State output.
OvrdIn	Override Input	Class: Digital - Initializes a timed override with each OFF-to-ON transition (OvrdEnb must also be ON). The override timer begins to count down after the Override Input returns to OFF or not active (NA).	—	If the Override Input and Override Cancel inputs are both at their high values, the Override Input value has precedence.
Cancel	Override Cancel	Class: Digital - Cancels a timed override with an ON. An OFF or not active (NA) has no effect on any override.	—	
OvrdTm	Override Time	Class: Analog - Defines the time period of a timed override, in minutes. A negative value, a value of 0.01, or a not active (NA) defaults as 0, resulting in no timed override.	0.02 to 240 minutes	—

Table–5.82 Control Override Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
State	Override State	Class: Digital - Indicates the present override status. This output defaults to not active (NA) whenever both the Input state and the Override Enable input are both NA.	OFF (0.0) ON (100.0)
TmRem	Time Remaining	Class: Analog - Indicates the remaining time in a timed override, in minutes. Any value greater than zero (> 0) indicates a timed override is in progress. This output defaults to not active (NA) whenever the Override Enable is NA.	0 to 240 minutes

Applying the Object

The signal at the Input is “or’d” with the timer override sequence to control the output. An ON at the Input always sets the Override State output to ON.

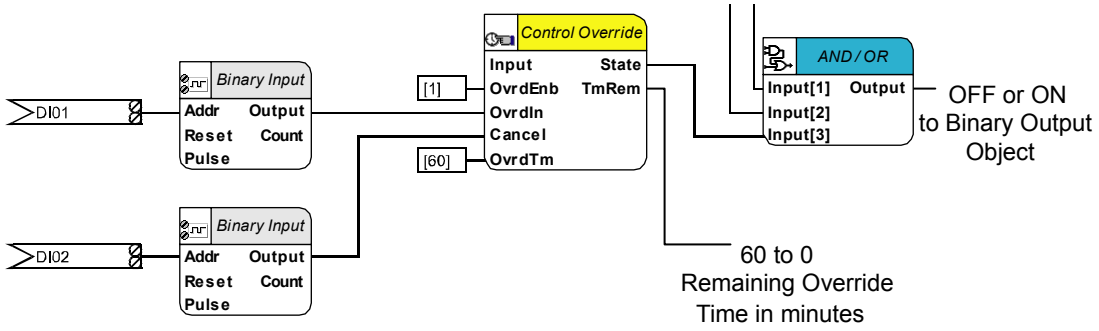
If the Input is OFF (or unconnected, not active (NA)), the other inputs for a timed override are evaluated. If enabled, a timed override is initialized with an OFF-to-ON transition at the Override Input, and begins to count down after this input returns to OFF or not active (NA). This count down lasts from 0.02 to 240 minutes, as determined by the Override Time input value.

A timed override also produces an analog value on the Time Remaining output, which counts down from the Override Time (1 to 240) to zero in whole minutes, when the timed override ends. At any time, an active timed override can be canceled by an ON to the Cancel input or be re-initiated by an OFF-to-ON transition at the Override Input.

The Override Cancel has no effect on the Override function whenever Override Input is activated and held in the ON condition.

Example Applications

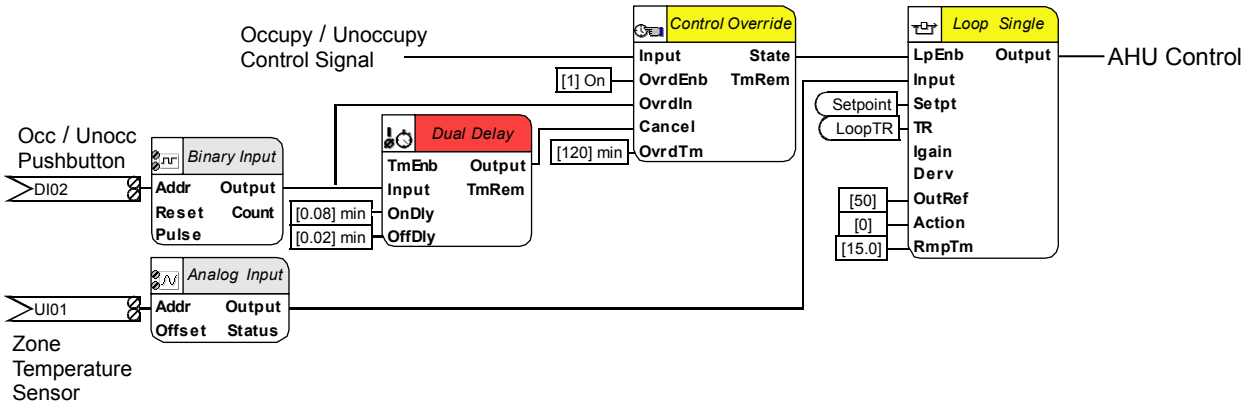
An example Control Override object below is configured for a timed override, with the Input left unconnected (NA) and a constant [1] fed to the Override Enable input. Two Binary Input objects representing momentary switches control the override. The Control Override object has a constant of 60 (minutes) as the Override Time. The Override State output from the Control Override object feeds the OR input of an AND / OR logic object, which in turn controls a Binary Output object (not shown), used to cycle OFF or ON an HVAC load.



Figure–5.45 Example Control Override Object used for a Timed Override.

In this application, any OFF-to-ON cycle at the OvrdIn input initializes a 60 minute override; any ON to the Cancel input terminates the timed override.

Figure–5.46 shows another Control Override object used for a timed override. In this case, a single momentary switch can both initiate the timed override and cancel it. This is accomplished using a Dual Delay object.



Figure–5.46 Control Override Object for Timed Override Using a Single Hardware Input.

The example above shows a portion of an air handler application in which a Loop object is enabled through the Control Override object. The occupy/unoccupy signal feeds the Input of the Control Override object, which is the main control enable signal. During the unoccupied period, the control function can be enabled by pressing the remote Occ/Unocc pushbutton. The control loop will be enabled for two hours. The override can be cancelled at any time by pressing and holding the pushbutton for at least 5 seconds.

Count Down

Object Usage: The Count Down object provides a means to count OFF-to-ON digital transitions on its Input, in a countdown fashion. This means the Total output decrements (by one) on each OFF-to-ON transition, starting from the assigned Count until 0 (zero) is reached. A Carry Flag output allows multiple Count Down objects to be cascaded for increased countdown capability. The Count Down object is similar to the Count Up object (page 200), which operates in a related but reverse (increment) mode.

Device Support: (all firmware revisions)

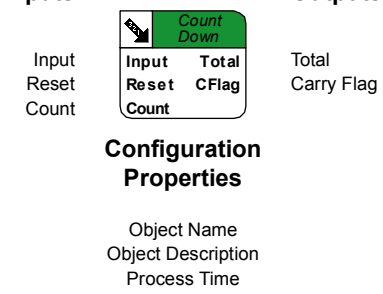
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 16 bytes (standard controllers)
6 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Properties

Table–5.83 Count Down Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.84 Count Down Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Digital - The Input tested for an OFF-to-ON transition, each of which causes the analog value at the Total output to be decremented by one.	—	A not active (NA) is ignored in the countdown algorithm.
Reset	Reset	Class: Digital - A digital ON to this input resets the Total output to the value present on the Count input and sets the Carry Flag output to OFF. A subsequent OFF to this input is required for the Input to be evaluated and the countdown algorithm to begin.	—	A not active (NA) is evaluated as an OFF.
Count	Count	Class: Analog - Defines the analog value that the Total output starts decrementing from until zero is reached and a rollover sequence begins.	1 to 10,000	A not active (NA) is evaluated the same as the maximum value (10,000).

Table–5.85 Count Down Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Total	Total	Class: Analog - Indicates the current count as the Count value minus the number of OFF-to-ON transitions until 0 (zero) is reached. The first transition following zero causes Total to be set to the Count value (rollover sequence). A Reset at any time also results in Total to be reset to the Count value.	0 to 10,000
CFlag	Carry Flag	Class: Digital - This output is set to ON for one count at a rollover sequence (the first Input OFF-to-ON after the Total output has counted down to zero).	OFF (0.0) ON (100.0)

Applying the Object

The countdown object monitors input state to implement the count down function. An OFF-to-ON or NA-to-ON transition causes the output total to decrease by one for each transition starting from the assigned count valve until zero is reached. The transition immediately after a zero count initiates a rollover sequence, where the object's Carry Flag output is set to ON and the Total output is set back to the Count input value. The next transition returns the Carry Flag output to OFF and restarts the count decrement on the Total output. The Carry Flag feature allows multiple Count Down objects to be cascaded for increased countdown capability.

A countdown sequence can be reset at any time with a digital ON on the Reset input, which sets the output Total equal to the Count value. Note that a subsequent OFF is required at the Reset input before the Count Down object begins again to countdown OFF-to-ON Input transitions.

The Count Down object is similar to the Count Up object, which operates in a related but reverse (increment) mode.

Count Up

Object Usage: The Count Up object provides a means to count OFF-to-ON digital transitions on its Input, in a count-up fashion. This means the Total output increments (by one) on each OFF-to-ON transition, starting at 0 (zero) until the assigned Count is reached. A Carry Flag output allows multiple Count Up objects to be cascaded for increased count-up capability. The Count Up object is similar to the Count Down object (page 198), which operates in a related but reverse (decrement) mode.

Device Support: (all firmware revisions)

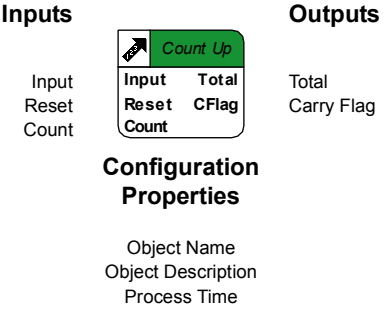
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 16 bytes (standard controllers)
6 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Properties

Table–5.86 Count Up Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.87 Count Up Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Digital - The Input tested for an OFF-to-ON or NA-to-ON transition, each of which causes the analog value at the Total output to be incremented by one.	—	OFF to not active (NA) is ignored in the count-up algorithm.
Reset	Reset	Class: Digital - A digital ON to this input: • Resets the Total output to 0 (zero). • Sets the Carry Flag output to OFF. A subsequent OFF to this input is required for the Input to be evaluated and the countup algorithm to begin.	—	A not active (NA) is evaluated as an OFF.
Count	Count	Class: Analog - Defines the analog value that the Total output must reach before a rollover sequence begins.	1 to 10,000	A not active (NA) is evaluated the same as the maximum value (10,000).

Table–5.88 Count Up Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Total	Total	Class: Analog - Indicates the current number of OFF-to-ON transitions since a 0 (zero) Total output. The first transition following a Total that equals the assigned Count causes Total to be set to zero (rollover sequence). A Reset at any time also results in Total to be reset to zero.	0 to 10,000
CFlag	Carry Flag	Class: Digital - This output is set to ON for one count at a rollover sequence (the first Input OFF-to-ON after the Total output has reached the Count).	OFF (0.0) ON (100.0)

Applying the Object

The Count Up object monitors a digital Input value and counts each state transition (OFF-to-ON or NA-to-ON occurrence) from 0 (zero), incrementing (by one) on each transition until the value of the assigned Count is reached. This count appears as an analog value at the object's Total output. The first transition after Total is equal to the assigned Count initiates a rollover sequence, where the object's Carry Flag output is set to ON and the Total output is set back to zero. The next transition returns the Carry Flag output to OFF and restarts the count increment on the Total output. The Carry Flag feature allows multiple Count Up objects to be cascaded for increased count-up capability.

A count-up sequence can be reset at any time with a digital ON on the Reset input, which sets the output Total equal to zero. Note that a subsequent OFF is required at the Reset input before the Count Up object begins again to count-up OFF-to-ON Input transitions.

The Count Up object is similar to the Count Down object, which operates in a related but reverse (decrement) mode.

COV Priority

(Rev.3 or Later Firmware Required)

Object Usage: The COV (Change of Value) Priority object has two primary inputs to receive values: Input[1] and Input[2]. In normal operation, the Enable input has a not active (NA) or ON and the Force[1] and [2] inputs have either an NA or OFF.

In this “normal mode”, the object outputs the last valid value received on either Input[1] or [2]. This is best described as “last-one-in goes out”. If the active input value changes to NA, the output falls back to the other input value, providing the input is still valid (and not NA). If a simultaneous value change occurs at both Inputs[1] and [2], Input[1] is given higher priority. If both Inputs[1] and [2] are NA, the output is set to the value at the Default input.

Force[1] and [2] inputs are digital inputs for bypassing the change-of-value function. A force input of ON “forces” the corresponding value at Input[1] or Input[2] to the Output, providing it is valid (not NA). If an ON is at both Force[1] and [2], the Force[1] action results. An OFF at the Enable input clears both internally stored values to NA and causes the value at the Default input to appear at the output. Whenever the Enable input is OFF, value changes at Inputs[1], [2] and Force[1], [2] are ignored. The Control Level output indicates by value (1 or 2) which input is currently in use. This output is set to 3 whenever the Default value is at the Output.

Device Support: (See page 7)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F2, F3, H2, H3, R2, R3, S1, S2, S3, or S4

MNL-11Rxx, -13Rxx
where xx = F2 or F3

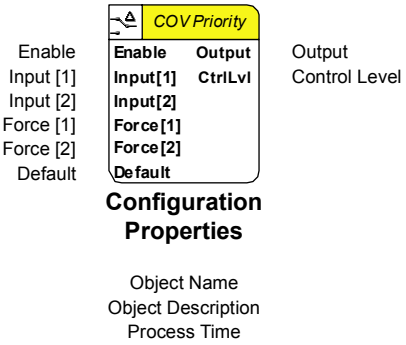
MNL-V1Rxx, -V2Rxx, -V3Rxx
where xx = V2 or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 16 bytes
RAM: 24 bytes (standard controllers)
8 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.89 COV Priority Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.90 COV Priority Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Enable	Enable	Class: Digital - A not active (NA) or ON enables the object (normal operation). An OFF causes the value at the Default input to go to the output, whereby the Control Level output is set to 3, indicating the Default value is used for control.	—	An OFF also clears the internal “scratch pad” stores for Inputs[1] and [2] to not active (NA).
Input[1]	Input [1]	Class: Analog - Input with the highest priority. • If a valid value, and the value has changed since the last execution, this value is passed to the output. The Control Level output is set to 1. • If the value has not changed since the last execution, Input[2] is evaluated.	-163.83 to 16383	A not active (NA) causes the object to evaluate Input[2].
Input[2]	Input [2]	Class: Analog - The input value with second priority. If this input is evaluated, the following sequence occurs: 1. If a valid value that has changed since the last execution, the value is passed to the output. The Control Level output is set to 2. 2. If a value not changed since the last execution, both outputs are not changed. 3. Not actives (NA) at this input and at Input[1] causes the value at the Default input to pass to the output. The Control Level output is set to 3. 4. An NA at this input, when the Control Level is 2 and when a valid value exists at Input[1], causes the output to be set to the value at Input[1]. The Control Level output is set to 1. 5. An NA at this input when the Control Level output is not 2 causes both outputs to remain at their current values.	-163.83 to 16383	Not active (NA) conditions are handled in the manner described in the sequence at left.

Table–5.90 COV Priority Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
Force[1]	Force [1]	Class: Digital - An ON forces the valid value at Input[1] to the Output, regardless of value change. Control Level output is set to 1. If Input[1] has a not active (NA), the Force[1] function is ignored.	—	Force[1] and Force[2] inputs are both ignored if the Enable input is OFF. Simultaneous ONs at both the Force[1] and Force[2] inputs result in the Force[1] condition.
Force[2]	Force [2]	Class: Digital - An ON forces the valid value at Input[2] to the Output, regardless of value change. Control Level output is set to 2. If Input[2] has a not active (NA), the Force[2] function is ignored.	—	
Default	Default	Class: Analog - Defines the default value passed to the output whenever both Inputs[1] and [2] have a not active (NA) or when the Enable input has an OFF.	-163.83 to 16383	Passes a not active (NA) if the input is unconnected or has an NA present.

Table–5.91 COV Priority Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output reflecting the Input[1] or [2] value that has changed most recently (or been forced via either force input). If Inputs[1] and [2] are both NA or the Enable input is OFF, the output is the value at the Default input.	-163.83 to 16383
CtrLvl	Control Level	Class: Analog - Identifies by number the active input. For example, 1 = Input[1], 2 = Input[2], 3 = Default input.	1, 2, or 3

Applying the Object

The COV Priority object allows a value to be overwritten by a newer value, automatically selecting from the two primary inputs: Input[1] and Input[2]. Consider the function between Inputs[1] and [2] as “last-one-in-wins”. Input[1] has a higher priority, meaning when a simultaneous value change occurs at both inputs, the Input[1] value is always passed.

Internally, the COV Priority object stores the last received value for each input in “scratch pad” memory. This allows the output to “fallback” to the stored value of the other input whenever the “active input” makes a transition from a valid value to a not active (NA). If both Inputs[1] and [2] have an NA, the output passes the value present at the Default input.

The Enable input must have a not active (NA) or ON for normal operation. An OFF at the Enable input clears both internally stored input values to NA and passes the Default value to the output. While the Enable input is OFF, all value changes at Inputs[1] and [2] are ignored.

Inputs Force[1] and [2] allow a method to “force” whatever valid value is at Input[1] or Input[2] to the output, disregarding past value changes. Both force inputs are digital, requiring an ON (value > 0) to force the corresponding input value. Force functions are ignored if an NA is at the respective Input[1] or [2], or if the object is not enabled (Enable = OFF).

The Control Level output indicates which input by number (1 or 2) is currently in use. If the Default input is in use, the Control Level output is 3.

Operation Sequence

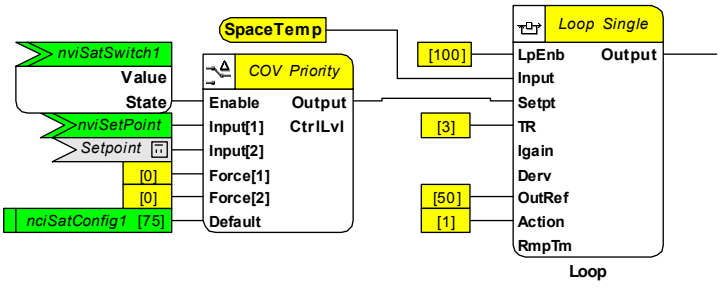
The COV Priority object operates by evaluating the inputs in this order upon *each cycle of execution*:

1. Enable input - If OFF, the output is set to the value at the Default input and the Control Level output is set to 3. Both internal input stores have NA. No further input evaluation occurs. If NA or ON, other inputs are evaluated in the order given below.
2. The Force[1] input is evaluated. If ON, any valid value at Input[1] is passed to the output, the Control Level output is set to 1, and no further input evaluation occurs. If Input[1] has an NA, the Force[1] function is ignored and input evaluation (below) continues. Evaluation also continues if Force[1] is OFF or NA (Force[1] function not active).
3. The Force[2] input is evaluated. If ON, any valid value at Input[2] is passed to the output, the Control Level output is set to 2, and no further input evaluation occurs. If Input[2] has an NA, the Force[2] function is ignored and input evaluation (below) continues. Evaluation also continues if Force[2] is OFF or NA (Force[1] function not active).
4. Input[1] - If a valid value is present, it is compared to previously stored value for Input[1]. If the valid value has changed, the output is set to this newer value and the Control Level output is set to 1. No further input evaluation occurs. If the value has not changed from the stored value, or if Input[1] has an NA, Input[2] is evaluated.
5. Input[2] - If a valid value is present, it is compared to previously stored value for Input[2]. If the valid value has changed, the output is set to this newer value and the Control Level output is set to 2. If the value has not changed from the stored value, the output and Control Level output remain at their current values.

Note: Whenever the object is enabled, values at Input[1] and Input[2] are written to their respective internal “scratch pad” upon each change.

Example

Following a controller reset, the COV Priority object below has an “NA” at the Enable input, which still allows it to operate. Until the bound NVIs receive valid values, it uses the setpoint value coming from the MN-Sx sensor. If the MN-Sx sensor value is also NA, it uses the value at the Default input.



Figure–5.47 COV Priority Object Used in Daily Setpoint Synchronization.

Under normal operation, the Enable input would be ON, allowing the COV Priority object to pass either Input[1] or Input[2] value on a “last-in” basis. The last input to change is sent to the “Output” and the “CtrlLvl” output updated accordingly. This allows the setpoint to be adjustable from either the operator PC or the local MN-Sx sensor. Additionally, by making the “Default” and “Enable” inputs accessible through the controller’s profile, the operator PC can gain sole control of the setpoint (by setting Enable input to OFF and adjusting the Default value as needed).

In this example, constant tags have been attached to the Force [1] and Force [2] inputs for diagnostic purposes from WP Tech. If desired, either or both of these inputs could instead be sourced from other control logic or made accessible via the controller’s profile.

Curve Fit

(Rev.3 or Later Firmware Required)

Object Usage: The Curve Fit object maps the analog value received on the Input to an x-y lookup table defined by its configuration properties. Up to six pairs of (x,y) data points provide up to five linear segments to approximate a desired curve. The output follows the piece-wise 'nonlinear' curve over the range defined in the properties. Additional inputs for Output Minimum and Maximum values are available to limit the output range.

Valid operation requires at least the first two pairs of data points (x1, y1) and (x2, y2) to be programmed in the configuration properties. Using additional objects, two or more Curve Fit objects can be cascaded to provide increased curve-fit resolution.

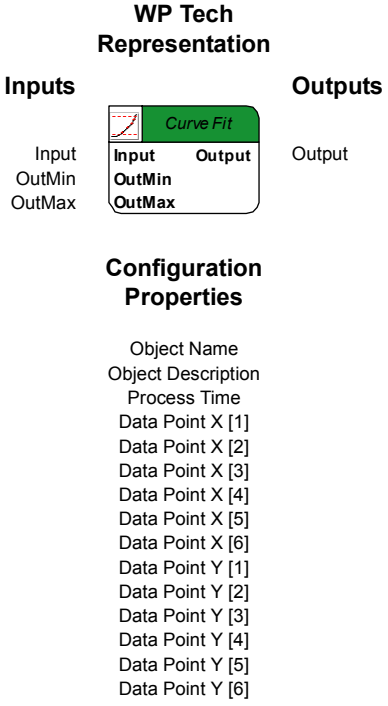
Device Support: (See page 7)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F2, F3, H2, H3, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MN 800 series

Memory Requirements: (per object)

EEPROM: 34 bytes
RAM: 38 bytes (standard controllers)
4 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Data Point X [1]
Data Point X [2]
Data Point X [3]
Data Point X [4]
Data Point X [5]
Data Point X [6]
Data Point Y [1]
Data Point Y [2]
Data Point Y [3]
Data Point Y [4]
Data Point Y [5]
Data Point Y [6]

WP Tech Stencil:
Logic and Math Control

Properties

Table–5.92 Curve Fit Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.92 Curve Fit Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
DataPtX1	Data Point X [1]	Class: Analog - Defines the first Input coordinate ‘x’, corresponding to the ‘y’ output of Data Point Y [1]	—	-163.83 to 16383	Mandatory. A not active (NA) sets the output to NA. Must be set to values in ascending order. Values X [1] and X [2] cannot be equal.
DataPtX2	Data Point X [2]	Class: Analog - Defines the second Input coordinate ‘x’, corresponding to the ‘y’ output of Data Point Y [2]	—		
DataPtX3	Data Point X [3]	Class: Analog - Defines the third Input coordinate ‘x’, corresponding to the ‘y’ output of Data Point Y [3]	—	-163.83 to 16383	Optional. X-Data points cannot be set to the same values. Must be in ascending order only. An out-of-sequence data point or one with a not active (NA) causes that data point and all further data points to be disregarded.
DataPtX4	Data Point X [4]	Class: Analog - Defines the fourth Input coordinate ‘x’, corresponding to the ‘y’ output of Data Point Y [4]	—		
DataPtX5	Data Point X [5]	Class: Analog - Defines the fifth Input coordinate ‘x’, corresponding to the ‘y’ output of Data Point Y [5]	—		
DataPtX6	Data Point X [6]	Class: Analog - Defines the sixth and last Input coordinate ‘x’, corresponding to the ‘y’ output of Data Point Y [6]	—		
DataPtY1	Data Point Y [1]	Class: Analog - Defines the first output coordinate ‘y’, corresponding to the ‘x’ input of Data Point X [1]	—	-163.83 to 16383	Mandatory. A not active (NA) sets the output to NA. May be set to values in ascending or descending order. Values Y [1] and Y [2] may be set to the same value.
DataPtY2	Data Point Y [2]	Class: Analog - Defines the second output coordinate ‘y’, corresponding to the ‘x’ input of Data Point X [2]	—		
DataPtY3	Data Point Y [3]	Class: Analog - Defines the third output coordinate ‘y’, corresponding to the ‘x’ input of Data Point X [3]	—	-163.83 to 16383	Optional. Adjacent Y-Data points can be set to any values including the same values (segment slope = 0), providing that the progression order is not reversed. An out-of-sequence data point or one with a not active (NA) causes that data point and all further data points to be disregarded.
DataPtY4	Data Point Y [4]	Class: Analog - Defines the fourth output coordinate ‘y’, corresponding to the ‘x’ input of Data Point X [4]	—		
DataPtY5	Data Point Y [5]	Class: Analog - Defines the fifth output coordinate ‘y’, corresponding to the ‘x’ input of Data Point X [5]	—		
DataPtY6	Data Point Y [6]	Class: Analog - Defines the sixth and last output coordinate ‘y’, corresponding to the ‘x’ input of Data Point X [6]	—		

Table–5.93 Curve Fit Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Analog - The input value to be characterized based on the curve defined by the configured x and y data points.	-163.83 to 16383	A not active (NA) sets the output to NA.
OutMin	Output Minimum	Class: Analog - Defines the minimum value allowed to be at the output. Typically less than the Output Maximum. If greater than the Output Maximum, the output goes to the Output Maximum.	-163.83 to 16383	If unconnected or not active (NA), the output is calculated without any minimum limit.
OutMax	Output Maximum	Class: Analog - Defines the maximum value allowed to be at the output. Typically greater than the Output Minimum. If less than the Output Minimum, the output goes to the Output Maximum.	-163.83 to 16383	If unconnected or not active (NA), the output is calculated without any maximum limit.

Table–5.94 Curve Fit Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - Reflects the calculated value based upon the input applied to the x-y curve defined by the data points (configuration properties). If the input value falls outside the range defined by the first and last valid X-data points, the output is calculated using the slope of the curve's segment that is closest to the input value. *Note:* The output is always limited by the values (if any) present at the inputs Output Minimum and Output Maximum.	-163.83 to 16383

Applying the Object

The Curve Fit object is useful in non-linear applications, common with pressure or flow type devices. The object produces an analog output that follows the input value applied to the x-y curve defined in the configuration properties. Input data is referenced to the x-axis and output data is referenced to the y-axis. The x-y curve can have up to five linear segments, based on up to six data (x,y) points.

The object requires at least two data points (x1, y1) and (x2, y2) defined in the configuration properties to produce a valid output. The object also requires that other optional data points are entered in a contiguous order, that is, proceeding with (x3, y3), (x4, y4), (x5, y5), and (x6, y6).

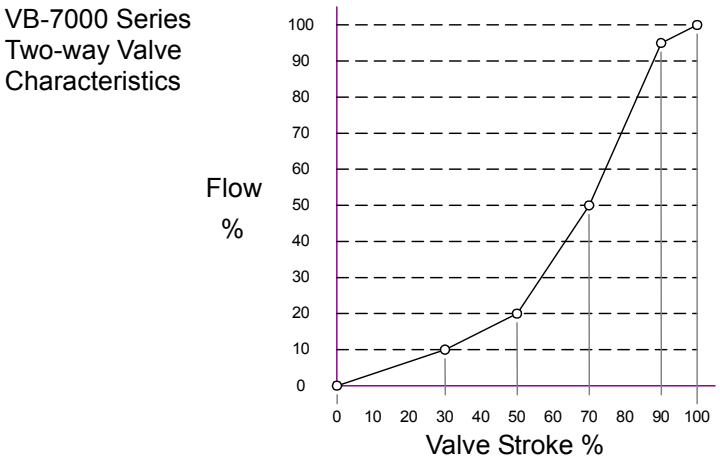
Examples

Three examples are included for the Curve Fit object:

- Example 1 - Valve Characterization (page 210)
- Example 2 - Curve Fit Object Cascade (page 212)
- Example 3 - Sensor Characterization (page 213)

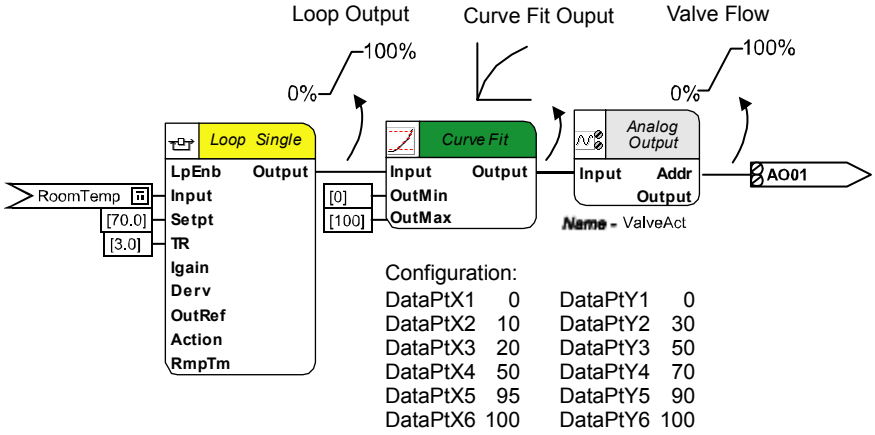
Example 1 - Valve Characterization

The following example illustrates a Curve Fit object used to generate a nonlinear output that matches the flow characteristics of a VB-7000 series valve. The Single Loop object generates a linear 0% and 100% output value based upon temperature and setpoint requirements. Operating the valve actuator directly from the loop output would cause the actual flow to follow the nonlinear curve as shown by the (VB-7000) Flow Characteristics Chart, Figure–5.48. A Single Loop output or actuator request of 50% would provide an actual valve flow of 20%.



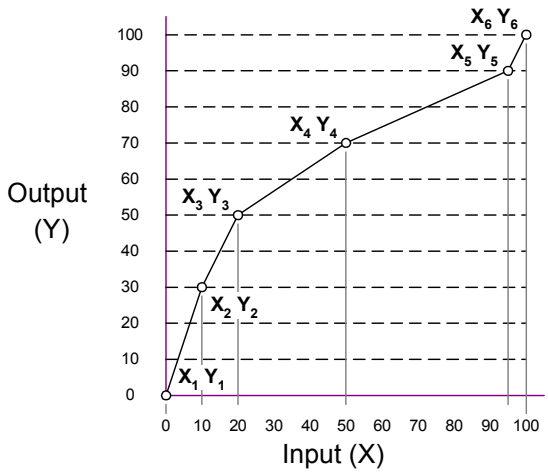
Figure–5.48 Example Stroke to Flow Characteristics for a VB-7000 Series Valve.

Adding a Curve Fit object between the Single Loop and Analog Output objects causes the loop output to be characterized, providing a nonlinear output to the valve actuator Figure-5.49. This allows the valve to operate based upon actual flow, producing an overall improvement in control.



Figure–5.49 Example Curve Fit Object Used to Linearize Actual Valve Flow.

The Curve Fit configuration data points are obtained using information found on the Flow Characteristics Chart (VB-7000 series in this example). The data point pairs are derived by transposing (X) and (Y) axis values that describe Valve Stroke vs. Valve Flow. This builds a nonlinear, piece-wise, five-segment curve to compensate for the valve's nonlinear characteristics, as shown in Figure–5.50 below.



Figure–5.50 Curve Fit Object Response (X-Y Curve) for Valve Example 1.

The result is a linear valve operation which allows the valve's actual flow to follow the Single Loop output request, as shown in Table–5.95 below. For example, a Single Loop output request of 50% now provides an actual valve flow of 50%.

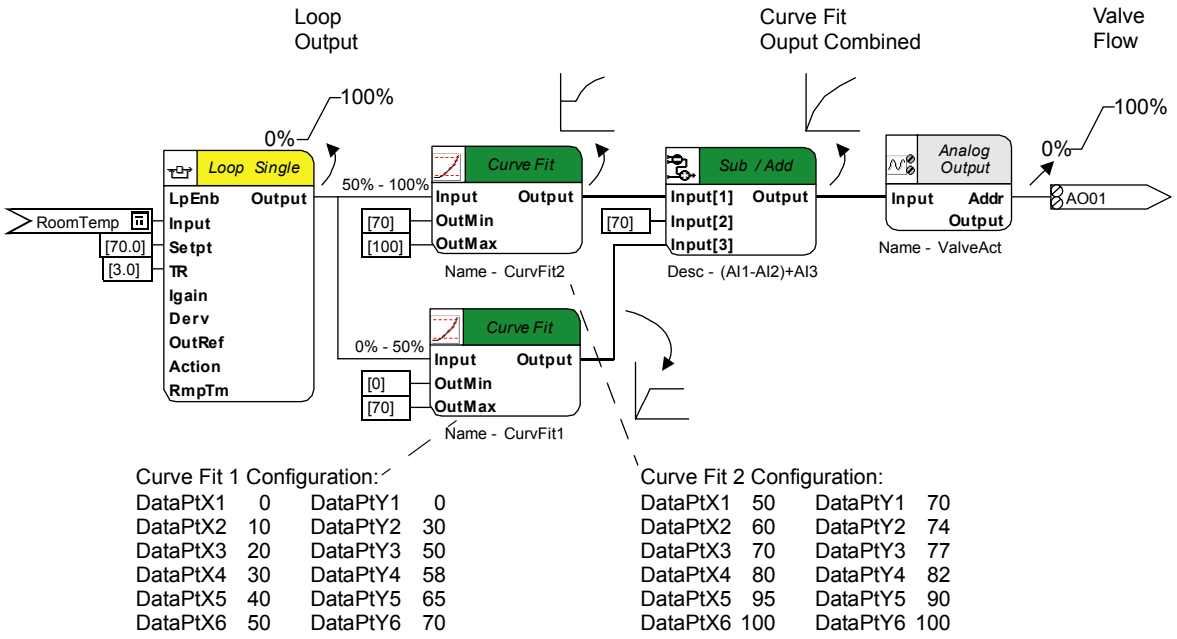
Table–5.95 Curve Fit Input-to-Output-to-Result Comparison, Valve Example.

Loop Output Curve Fit Input (X)	Curve Fit Output (Y) Actuator Stoke	Result Actual Valve Flow
0%	0%	0%
10%	30%	10%
20%	50%	20%
50%	70%	50%
95%	90%	95%
100%	100%	100%

Note: This example is simplified for the purpose of describing the Curve Fit function. An application may not necessarily compensate for the flow characteristics of the valve itself. A typical heating application might use the Curve Fit object to characterize the stroke of a valve to the amount of energy (BTUs, kJ) generated by the valve control of the mechanical equipment.

Example 2 - Curve Fit Object Cascade

This example is based on the previous valve characterization example, but is expanded to show how two Curve Fit objects can be cascaded to increase the resolution of the x-y curve. In this case, additional (x,y) data points are obtained using the same (VB-7000) series brass valve Flow Characteristics Chart. As in the previous example, data point pairs are derived by transposing (X) and (Y) axis values which describe Valve Stroke vs. Valve Flow. A higher resolution nonlinear curve results with a ten segment curve.

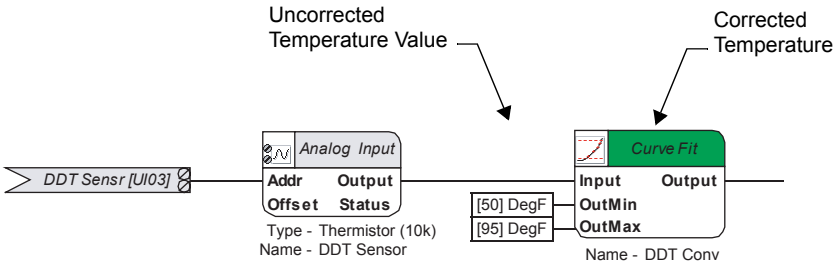


Figure–5.51 Example of Cascading Curve Fit Objects to Linearize Actual Valve Flow.

In this example, Curve Fit 1 is used to characterize loop output (X) values between 0% and 50%. Curve Fit 2 is used to characterize loop output (X) values between 50% and 100%. The outputs of both Curve Fit objects are combined using a Sub / Add object to generate the higher resolution curve, thus improving flow control resolution.

Example 3 - Sensor Characterization

A Curve Fit object can be used to “normalize” the response of a nonstandard 10k thermistor RTD sensor that has a similar characteristic to the standard 10k thermistor with 11k shunt. In this scenario, one of the Satchwell 10k thermistor sensors (models DUT, DRT, DDT, DST, or DWT) must be used with a universal input (UI) and Analog Input object. Without the Curve Fit object, the value produced by the Analog Input object will be in error.



Curve Fit Configuration:

DataPtX1	25.9	DataPtY1	14.0
DataPtX2	40.0	DataPtY2	32.0
DataPtX3	55.9	DataPtY3	50.0
DataPtX4	81.7	DataPtY4	77.0
DataPtX5	181.0	DataPtY5	176.0

Figure–5.52 Curve Fit Object Used for Sensor Characterization.

Using the sensor characteristics provided in the Satchwell documentation, equivalent resistance values were applied to an Analog Input object (set as Thermistor (10k)). Temperature values at each point were recorded, as shown in Table–5.96 (metric units) and Table–5.97 (English units).

Table–5.96 Resistance / Temperature, Satchwell 10k Thermistor (°C).

Temperature (°C)	Resistance (ohms)	Measured (°C)	Temperature (°C)	Resistance (ohms)	Measured (°C)
-10	8471	-3.4	60	2055	63.1
-5	8093	0.4	65	1791	68.2
0	7661	4.5	70	1562	73.3
5	7182	8.8	75	1363	78.3
10	6669	13.2	80	1193	83.3
15	6126	17.9	85	1047	88.3
20	5573	22.7	90	921	93.1
25	5025	27.6	95	814	97.9
30	4492	32.6	100	721	102.6
35	3989	37.6	105	642	106.9
40	3518	42.7	110	574	111.1
45	3089	47.7	115	516	115.1
50	2702	52.9	120	466	119
55	2358	58			

Table–5.97 Resistance / Temperature, Satchwell 10k Thermistor (°F).

Temperature (°F)	Resistance (ohms)	Measured (°F)	Temperature (°F)	Resistance (ohms)	Measured (°F)
14	8471	25.9	140	2055	145.6
23	8093	32.6	149	1791	154.8
32	7661	40.0	158	1562	163
41	7182	47.8	167	1363	172
50	6669	55.9	176	1193	181
59	6126	64.3	185	1047	190
68	5573	72.9	194	921	199
77	5025	81.7	203	814	208
86	4492	90.6	212	721	216
95	3989	99.7	221	642	224
104	3518	108.8	230	574	233
113	3089	118.0	239	516	240
122	2702	127.3	248	466	247
131	2358	136.4			

In this example, the output of the Analog Input object feeds the input of a Curve Fit object. Curve Fit object configuration data point pairs are entered using the data obtained in the table. Six data points are chosen to represent the best possible fit of a nonlinear, five-segment curve to compensate for the nonstandard 10k thermistor characteristics (see shaded cells in Table–5.96 and Table–5.97, above). ***X-values*** are entered using the *measured* temperature values. ***Y-values*** are entered using the *required* temperature values. The output of the Curve Fit object produces the correct (compensated) temperature value.

Demux Select

(Rev.3 or Later Firmware Required)

Object Usage: The Demux Select object provides a means for “demultiplexing” or directing the input value to one of six individual outputs. An output is “chosen” by the value received at the Select input (number from 1 to 6). Outputs “not chosen” produce their corresponding default values, which are available as inputs to the object. Default values can be any values including not active (NA).

The Select input evaluates only integer values received between the range of 1 and 6. Decimal portions of numbers in this range are ignored, for example, 4.77 at the Select input is processed as 4 and 6.93 is processed as 6. Any “out-of-range” Select input value (such as NA or 7) causes all Outputs[1] to [6] to pass their associated default values.

Device Support: (See page 7)

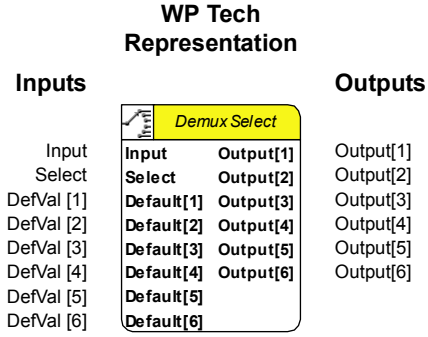
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F2, F3, H2, H3, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx
where xx = V2 or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 20 bytes
RAM: 32 bytes (standard controllers)
12 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.98 Demux Select Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.99 Demux Select Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Analog - The main input value, passed to one of the six outputs whenever the Select input has a value from 1 to 6.	-163.83 to 16383	
Select	Select	Class: Analog - Evaluated as follows: • A value of 0 sets all Outputs[1] to [6] to the values present at the associated Default Value[1] to [6] inputs. • A value from 1 to 6 selects the corresponding Output[1] to [6] to pass the main input value. All other outputs are set to the values present at the associated Default Value[1] to [6] inputs.	0 to 6	A not active (NA), negative value, or other value outside the normal range (7 or above) is evaluated the same as 0. Decimal portions are truncated in the evaluation. For example, a value of 3.54 is evaluated as 3.
DefVal1	Default Value [1]	Class: Analog - The value passed to Output[1] whenever the Select value is NOT equal to 1.	-163.83 to 16383	Default Value inputs can be any value or not active (NA).
DefVal2	Default Value [2]	Class: Analog - The value passed to Output[2] whenever the Select value is NOT equal to 2.	-163.83 to 16383	
DefVal3	Default Value [3]	Class: Analog - The value passed to Output[3] whenever the Select value is NOT equal to 3.	-163.83 to 16383	
DefVal4	Default Value [4]	Class: Analog - The value passed to Output[4] whenever the Select value is NOT equal to 4.	-163.83 to 16383	
DefVal5	Default Value [5]	Class: Analog - The value passed to Output[5] whenever the Select value is NOT equal to 5.	-163.83 to 16383	
DefVal6	Default Value [6]	Class: Analog - The value passed to Output[6] whenever the Select value is NOT equal to 6.	-163.83 to 16383	

Table–5.100 Demux Select Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Analog - Reflects the value of either the main input or Default Value [1] input, depending on the value at the Select input.	-163.83 to 16383
Output[2]	Output [2]	Class: Analog - Reflects the value of either the main input or Default Value [2] input, depending on the value at the Select input.	-163.83 to 16383
Output[3]	Output [3]	Class: Analog - Reflects the value of either the main input or Default Value [3] input, depending on the value at the Select input.	-163.83 to 16383
Output[4]	Output [4]	Class: Analog - Reflects the value of either the main input or Default Value [4] input, depending on the value at the Select input.	-163.83 to 16383
Output[5]	Output [5]	Class: Analog - Reflects the value of either the main input or Default Value [5] input, depending on the value at the Select input.	-163.83 to 16383
Output[6]	Output [6]	Class: Analog - Reflects the value of either the main input or Default Value [6] input, depending on the value at the Select input.	-163.83 to 16383

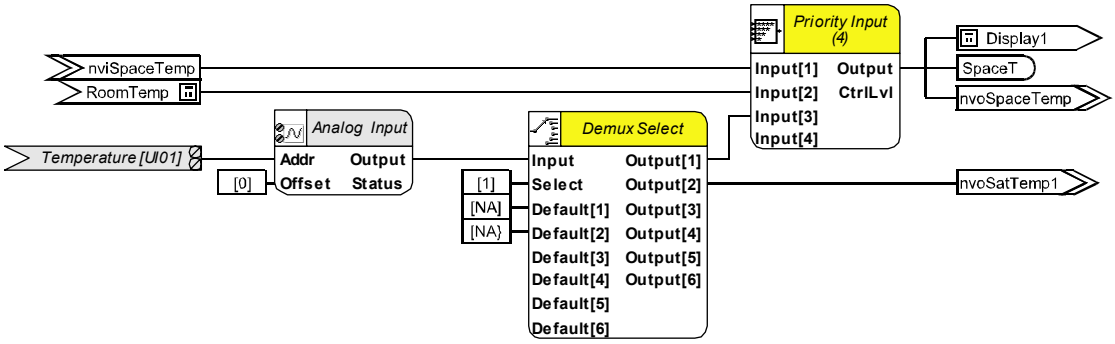
Applying the Object

The Demux Select object “demultiplexes” the single main input. This means the input value can be passed to ONE of the Outputs[1] to [6], depending on the value (1 to 6) at the Select input. Outputs not selected to pass the input value pass instead the value at their associated Default Value[1] to [6] input.

Logically, the object functions as a single-pole, seven-throw switch, with one selection being “OFF” (main input not passed). This occurs when the Select input value is 0, not active (NA), or an out-of-range value (not 0 to 6). In this case, all outputs are set to value at their associated Default Value input.

Example

The Demux object is useful when engineering an application that can be used in a variety of situations. Figure–5.53 below shows the Demux object used to select the use of a sensor attached to the controller’s UI 1 input.



Figure–5.53 Demux Object Used For Application Versatility.

In this example, the room temperature (control point) can be from either a bound NVI input, MN-Sx sensor, or standard sensor connected to UI 1, courtesy of the Priority Input (4) object. The addition of the Demux object between the Analog Input object (for the sensor at UI 1) and Input[3] of the Priority Input (4) object allows flexibility for how the sensed value is used.

For example, if the application’s control point (SpaceT) is to come from the MN-Sx sensor (RoomTemp tag), and the sensor at UI 1 is needed for another temperature, the programmer can set the “Select” input of the Demux Select object to “2”, instead of the “1” shown. This redirects the sensor value to the more generic “nvoSatTemp1” output of the profile, and outputs the default “NA” value to the Priority Input (4) object.

Dual Delay

Object Usage: The Dual Delay object monitors a digital Input and provides a delayed digital Output response. Both OFF-to-ON input transitions (On Delay) and ON-to-OFF input transitions (Off Delay) are monitored, each with specified delay times from 0.0 to 1,000.0 minutes. A Time Remaining output provides the current remaining minutes in any active On Delay or Off Delay. The dual delay function can be disabled with an OFF at the Time Enable input, which causes the Output to directly track the Input state. A not active (NA) to the Input is evaluated as an OFF.

The Dual Delay object combines the functions available separately in the On Delay (page 355) and Off Delay (page 352) objects.

Digital Output = Digital Input
(following the On Delay or Off Delay)

Device Support: (all firmware revisions)

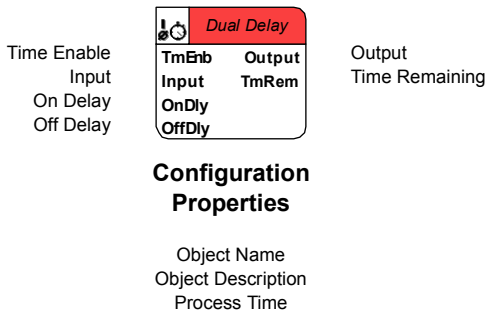
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 12 bytes
RAM: 20 bytes (standard controllers)
8 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Timer and Sequence Control

Reference Listing of All Timer Objects

Object Name	Digital Input to Digital Output Behavior
Dual Delay	Both an On Delay and an Off Delay
Dual Minimum	Both Minimum ON and Minimum OFF
Minimum On	Minimum ON period before OFF
Minimum Off	Minimum OFF period before ON
On Delay	Delay before Output ON
Off Delay	Delay before Output OFF

Properties

Table–5.101 Dual Delay Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.102 Dual Delay Object Input Properties.

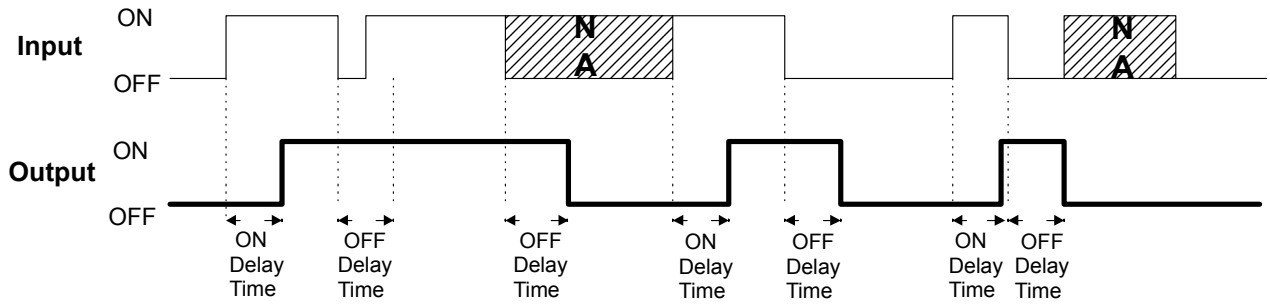
Abbrev.	Name	Class / Description	Range / Selection	Notes
TmEnb	Time Enable	Class: Digital - An ON or not active (NA) enables the dual delay function. An OFF at this input disables the dual delay function, causing the Output to directly follow the Input (no delays).	—	
Input	Input	Class: Digital - The input signal to which the dual delay is applied. An NA is evaluated as OFF.	—	See the Timing Diagram for Input to Output operation.
OnDly	On Delay	Class: Analog - The value of the ON delay time in minutes. A negative or not active (NA) value disables the delay as 0.0 minutes.	0.0 to 1,000.0 minutes	Decimal values are valid. For example, 0.25 is evaluated as 15 seconds.
OffDly	Off Delay	Class: Analog - The value of the OFF delay time in minutes. A negative or not active (NA) value disables the delay as 0.0 minutes.	0.0 to 1,000.0 minutes	Decimal values are valid. For example, 0.25 is evaluated as 15 seconds.

Table–5.103 Dual Delay Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The Output is set to match the Input state following a delay timer sequence (On Delay or Off Delay), or if the TmEnb input is OFF.	OFF (0.0) ON (100.0)
TmRem	Time Remaining	Class: Analog - The analog value representing the amount of active ON delay or OFF delay time (in whole minutes).	0 to 1,000 minutes

Applying the Object

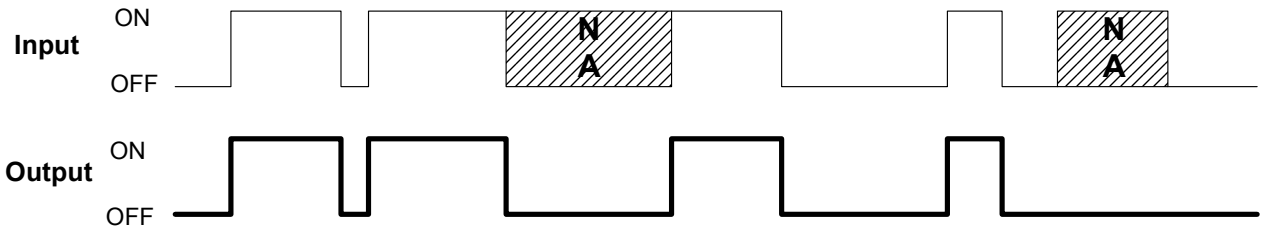
The Dual Delay object allows a time-delayed response for any change to a digital signal (monitored on the object's Input). The digital Output provides both an ON delay or an OFF delay based on the transition received at the Input (OFF-to-ON or ON-to-OFF). An ON delay or OFF delay can last from 0.1 to 1000.0 minutes, based on the value present at the separate On Delay and Off Delay inputs. The Time Enable input must be at ON or not active (NA) to provide delays. Figure–5.54 shows the function of a Dual Delay object for both an on delay and off delay.



Figure–5.54 Timing Diagram for a Dual Delay Object with the Delay Function Enabled (Time Enable input is ON or NA).

During an active ON delay or OFF delay, the Time Remaining output is an analog value for the current remaining delay time (in whole minutes). This value counts down each minute as the delay timer expires, where the Time Remaining output is at 0 (zero) and the Output goes to the same digital state as the Input.

The Dual Delay function is disabled while an OFF is at the Time Enable input. In this condition, the Output directly tracks the Input without delay as shown in Figure–5.55, and the Time Remaining output remains at 0 (zero).



Figure–5.55 Timing Diagram for a Dual Delay Object with the Delay Function Disabled (Time Enable input is OFF).

Note: After a controller reset the object operates as if the input and output were off prior to the reset.

Dual Minimum

Object Usage: The Dual Minimum object monitors a digital Input and prevents the Output from being set to digital OFF or digital ON before a specified minimum timeout has expired. Timeouts are defined by Minimum On and Minimum Off inputs to the object, and can range from 0.0 to 1,000.0 minutes. A Time Remaining output provides the current remaining minutes in any active Minimum On or Minimum Off period. The Dual Minimum function can be disabled with an OFF at the Time Enable input, which causes the Output to directly track the Input state. A not active (NA) to the Input is evaluated as an OFF.

The Dual Minimum object combines the functions available separately in the Minimum On (page 342) and Minimum Off (page 339) objects.

Device Support: (all firmware revisions)

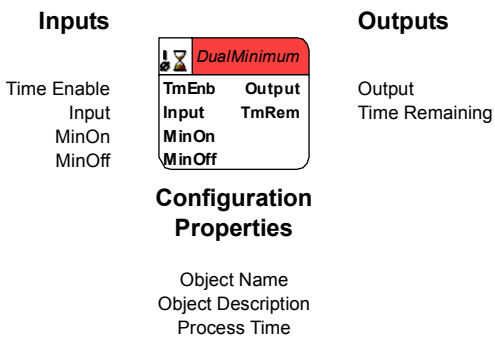
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 12 bytes
RAM: 20 bytes (standard controllers)
8 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Timer and Sequence Control

Reference Listing of All Timer Objects

Object Name	Digital Input to Digital Output Behavior
Dual Delay	Both an On Delay and an Off Delay
Dual Minimum	Both Minimum ON and Minimum OFF
Minimum On	Minimum ON period before OFF
Minimum Off	Minimum OFF period before ON
On Delay	Delay before Output ON
Off Delay	Delay before Output OFF

Properties

Table–5.104 Dual Minimum Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.105 Dual Minimum Object Input Properties.

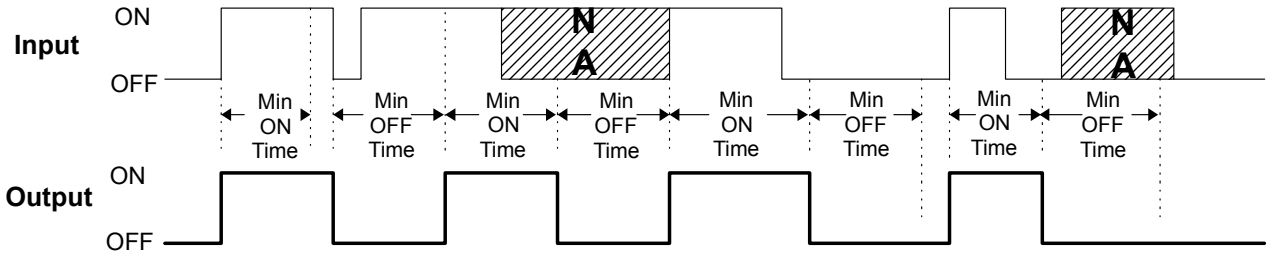
Abbrev.	Name	Class / Description	Range / Selection	Notes
TmEnb	Time Enable	Class: Digital - An ON or not active (NA) enables the Dual Minimum timeout function. An OFF at this input disables the Dual Minimum timeout function, causing the Output to directly follow the Input.	—	
Input	Input	Class: Digital - The input signal to which the Dual Minimum function is applied. An NA is evaluated as OFF.	—	See the Timing Diagram for Input to Output operation.
MinOn	Minimum On Time	Class: Analog - The value of timeout (in minutes) for the Minimum On period. A negative or not active (NA) value disables the Minimum On timeout as 0.0 minutes.	0.0 to 1,000.0 minutes	Decimal values are valid. For example, 0.25 is evaluated as 15 seconds.
MinOff	Minimum Off Time	Class: Analog - The value of timeout (in minutes) for the Minimum Off period. A negative or not active (NA) value disables the Minimum Off timeout as 0.0 minutes.	0.0 to 1,000.0 minutes	Decimal values are valid. For example, 0.25 is evaluated as 15 seconds.

Table–5.106 Dual Minimum Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The Output is set to match the Input state following completion of the appropriate Minimum On or Minimum Off timeout, or if the TmEnb input is OFF.	OFF (0.0) ON (100.0)
TmRem	Time Remaining	Class: Analog - The analog value representing the amount of active Minimum On or Minimum Off time remaining (in whole minutes).	0 to 1,000 minutes

Applying the Object

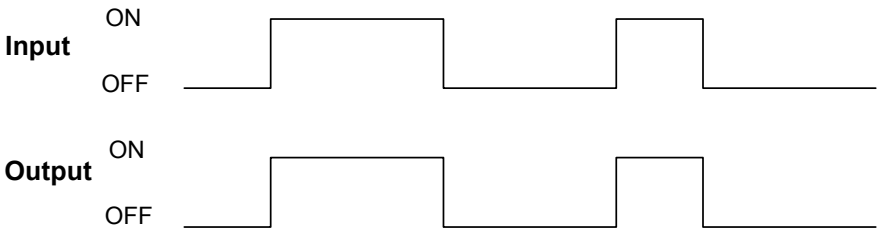
The Dual Minimum object prevents short-cycling of a digital Output by using minimum ON and minimum OFF timeouts applied to Input and Output changes. Timeouts are defined by Minimum On and Minimum Off inputs to the object, and can range from 0.0 to 1,000.0 minutes. The Time Enable input must be at ON or not active (NA) to provide Minimum timeouts. The timing diagram in Figure–5.56 shows Dual Minimum object operation for both Minimum On and Minimum Off times.



Figure–5.56 Timing Diagram for a Dual Minimum Object with Minimum Times Enabled (Time Enable = ON or NA).

During an active Minimum On or Minimum Off timeout period, the Time Remaining output is the analog value for the current remaining timeout (in whole minutes). This value counts down each minute as the timeout expires, where the Time Remaining output is at 0 (zero) and the Output goes to the same digital state as the Input.

The Dual Minimum function is disabled while an OFF is at the Time Enable input. In this condition, the Output directly tracks the Input as shown in Figure–5.57, and the Time Remaining output remains at 0 (zero).

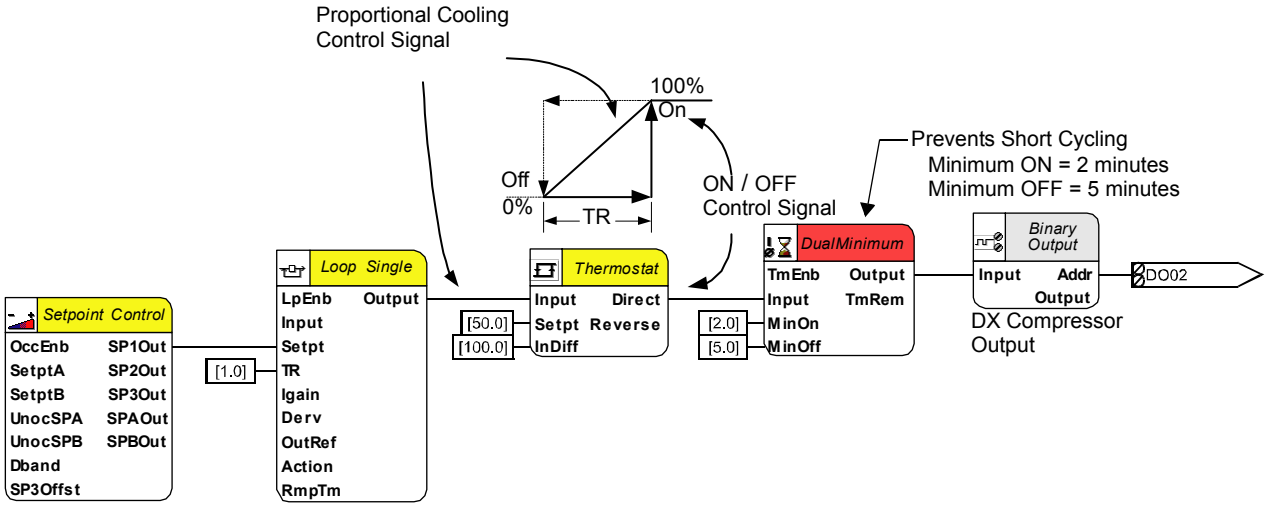


Figure–5.57 Timing Diagram for a Dual Minimum Object with Minimum Times Disabled (Time Enable input is OFF).

Note: After a controller reset the object operates as if the input and output were off prior to the reset.

Example

Figure–5.58 shows a Dual Minimum object used for short-cycle protection of a direct expansion (DX) compressor in a cooling RTU application. Both ON and OFF protection is provided as the output of the Thermostat object changes in response to the Loop object output.



Figure–5.58 Dual Minimum Object Used for Short Cycle Protection of a DX Compressor.

DUI Expander

(Rev.3 or Later Firmware Required)

Object Usage: The DUI (Digital Universal Input) Expander object is a point-type object that provides support for up to five digital inputs using a single controller UI (universal input). Each DUI Expander requires a specific resistor-to-resistor (R2R) network connected to the UI for further connection to the various field contacts. The object monitors the binary (OFF or ON) status of each of the five R2R inputs, which are individually configurable as either normally open or closed (direct or reverse). Only dry (voltage-free) contacts can be monitored. Status of each input is represented in the DUI Expander object by digital Outputs[1] through [5]. An additional Status Flags output produces an enumerated value if an under- or over-range condition occurs, or if the object is improperly setup.

Note: Pulse-rate and count functions are not available in the DUI Expander object. A contact switching at a rate that exceeds 0.2 Hz (50% duty cycle) may not be conveyed to the associated DUI output. These functions require a Binary Input object.

Internally, the DUI Expander function relies upon the total measured resistance value. Proper operation is assured only if the associated R2R resistor network is built to supply the resistance values as outlined in Table–5.114 and Table–5.115. The length of 18-gauge wire used between the switch terminals and the dry contact switches should not exceed 100 feet (30.4 m). Similarly, 18-gauge wire used to connect the controller and the DUI must not exceed a length of 100 feet (30.4 m). The contact resistance of the dry contact closures must not exceed 1 ohm. An open switch contact connected to the DUI R2R resistor network should have a minimum resistance of 1 megohm. After switching, the dry contact resistance must stabilize within 100 milliseconds. The dry contact(s) selected must be capable of low current (3.1 mA) operation.

Device Support: (See page 7)

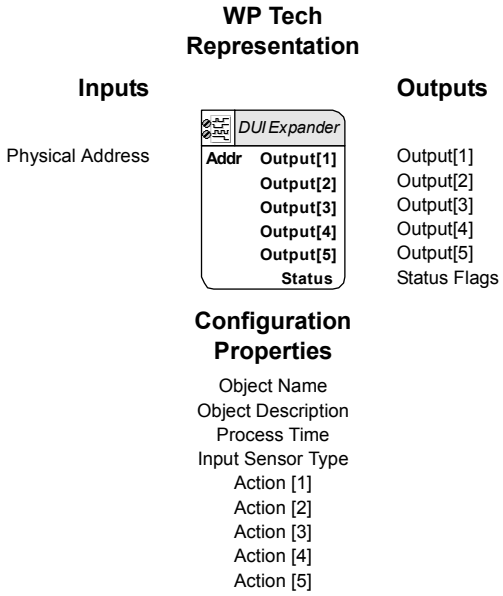
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F2, F3, H2, H3, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx
where xx = V2 or V3

MN 800 series

Memory Requirements: (per object)

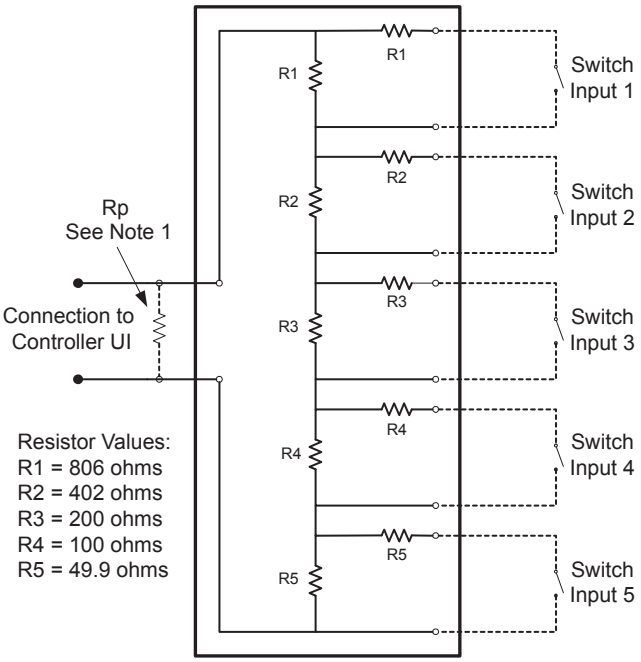
EEPROM: 18 bytes
RAM: 30 bytes (standard controllers)
12 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
IO and Alarm Control

Required R2R Resistor Network



Note 1: Rp = 22.1K ohms 1%
Parallel resistor required for interface to the MN 800 controller only.

See Table–5.110 for complete details.

Properties

Table–5.107 DUI Expander Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
Type	Input Sensor Type	Class: Analog - Defines the sensor type connected to the hardware input. Must be set to Resistance 1k Switched (8) for the DUI Expander function. If set to Disabled (0), no DUI Expander occurs.	0	0 - Disabled 8 - Resistance 1k Switched (Normal)	Not Active or values outside valid range defaults to 0 (Disabled).
Action[1]	Action [1]	Class: Analog - Defines the action of Output[1] - (0 = Direct and 1 = Reverse).	0	0 - Direct 1 - Reverse	If Direct (0), a contact closure evaluates ON, and a contact open as OFF. (Normally Open) If Reverse (1), a contact closure evaluates OFF, and a contact open as ON. (Normally Closed)
Action[2]	Action [2]	Class: Analog - Defines the action of Output[2] - (0 = Direct and 1 = Reverse).	0	0 - Direct 1 - Reverse	
Action[3]	Action [3]	Class: Analog - Defines the action of Output[3] - (0 = Direct and 1 = Reverse).	0	0 - Direct 1 - Reverse	
Action[4]	Action [4]	Class: Analog - Defines the action of Output[4] - (0 = Direct and 1 = Reverse).	0	0 - Direct 1 - Reverse	
Action[5]	Action [5]	Class: Analog - Defines the action of Output[5] - (0 = Direct and 1 = Reverse).	0	0 - Direct 1 - Reverse	

Table–5.108 DUI Expander Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Addr	Physical Address	Class: Analog - Indicates the physical hardware address (input terminal point on the controller) assigned to the DUI Expander object. Only a UI (Universal Input) can be used.	Dependent on the controller platform selected.	If no physical hardware address is assigned (NA), Outputs[1] to [5] are set to NA and the Status Flags output is set to 1.

Table–5.109 DUI Expander Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital - Reflects the state of Input 1 on the R2R resistor-network. • Direct Action: OFF at contact open, ON at contact close. • Reverse Action: ON at contact open, OFF at contact close.	Direct or Reverse: OFF (0) ON (100)

Table–5.109 DUI Expander Object Output Properties. (Continued)

Abbrev.	Name	Class / Description	Valid Values
Output[2]	Output [2]	Class: Digital - Reflects the state of Input 2 on the R2R resistor-network. • Direct Action: OFF at contact open, ON at contact close. • Reverse Action: ON at contact open, OFF at contact close.	Direct or Reverse: OFF (0) ON (100)
Output[3]	Output [3]	Class: Digital - Reflects the state of Input 3 on the R2R resistor-network. • Direct Action: OFF at contact open, ON at contact close. • Reverse Action: ON at contact open, OFF at contact close.	Direct or Reverse: OFF (0) ON (100)
Output[4]	Output [4]	Class: Digital - Reflects the state of Input 4 on the R2R resistor-network. • Direct Action: OFF at contact open, ON at contact close. • Reverse Action: ON at contact open, OFF at contact close.	Direct or Reverse: OFF (0) ON (100)
Output[5]	Output [5]	Class: Digital - Reflects the state of Input 5 on the R2R resistor-network. • Direct Action: OFF at contact open, ON at contact close. • Reverse Action: ON at contact open, OFF at contact close.	Direct or Reverse: OFF (0) ON (100)
Status	Status Flags	Class: Analog - Indicates an error condition if a non-zero value, as follows: 0 - Valid setup and normal object operation. 1 - Physical address set to not active (NA). 2 - Under-range condition (total resistance less than allowed). 3 - Over-range condition (total resistance more than allowed).	0, 1, 2, or 3

Applying the Object

The DUI Expander object allows a single universal input (UI) of an I/A Series MicroNet standard controller (Rev. 3 or higher firmware) or an MN 800 controller to monitor the status of up to five dry contacts.

Caution:

- In all Universal Inputs, noise can cause erratic and erroneous UI operation. To avoid these issues, proper precautions must be taken during the wiring process. See the *I/A Series MicroNet System Engineering Guide*, F-26507, for wiring details.
- In addition, unstable (fluttering) contact closures or contacts that exhibit varying resistances not within the specified tolerances, will cause the DUI outputs to reflect erratic and erroneous digital states. Proper precautions must be taken to ensure that the dry contact inputs remain stable and switch within the listed contact tolerances.

Each DUI Expander object requires one specific R2R resistor network, as shown in Table–5.110 below.

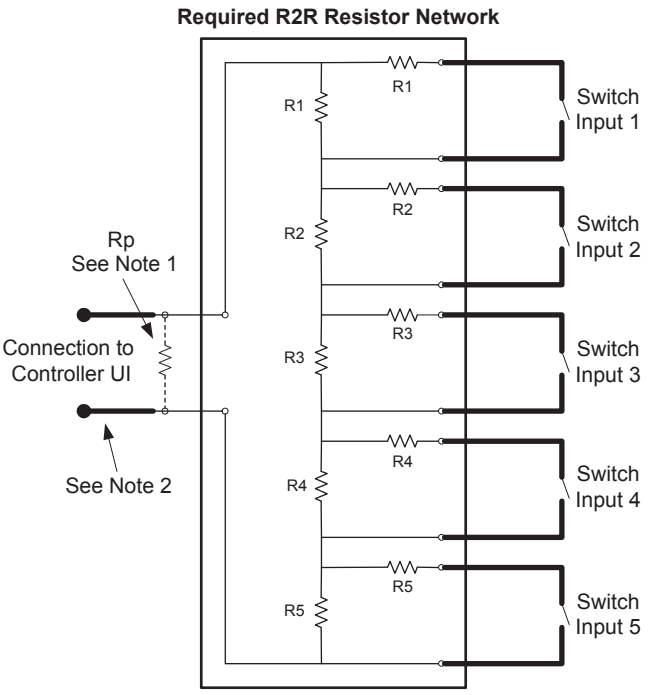
Table–5.110 R2R Resistor Network Required by the DUI Expander Object.

Resistor	Value	Needed
R1	806 Ω	2
R2	402 Ω	2
R3	200 Ω	2
R4	100 Ω	2
R5	49.9 Ω	2

Notes

- The DUI module used must be designed to provide the nominal equivalent resistances shown in Table–5.114 (without parallel resistor Rp) and Table–5.115 (with parallel resistor Rp required for the MN 800 controller).
- The contact resistance of the dry contact closures must not exceed 1 ohm. An open switch contact connected to the DUI R2R resistor network should a have a minimum resistance of 1 megohm. After switching, the dry contact resistance must stabilize within 100 millisecond. The dry contact must be sized to handle a 3.1 mA current source.
- The length of 18-gauge wire used between the switch terminals and the dry contact switches should not exceed 100 feet (30.4 m). Similarly, 18-gauge wire used to connect the controller and the DUI must not exceed a length of 100 feet (30.4 m).
- Exceeding the maximum contact resistance, the wire length specifications, or the dry contact equivalent resistances listed in Table–5.114 and Table–5.115 could cause the controller to erroneously read the digital input states.
- See Table–5.114 and Table–5.115, detailing the resistance values for all input contact combinations.

Diagram



Note 1: Rp = 22.1K ohms 1%
Parallel resistor required for interface to the MNL-800 controller only.

Note 2: Each of the twelve external wire lengths highlighted in bold must be evenly distributed and limited to 100 feet (18 AWG) maximum.

Contacts monitored by the R2R inputs may be any combination of normally open (N.O.) or normally closed (N.C.) contacts. Contact type by R2R input must be configured using Action[1] through [5] configuration properties.

Action Configuration

The Action[1] through Action[5] configuration properties determine the action of the Output[1] through Output[5] in relation to the digital conditions at switch inputs 1 through 5 of the R2R resistance network. Action[x] properties can be set in any combination required to support the field contacts, where:

- Direct (0) causes the output value to directly reflect the digital condition of the input at the R2R resistance network. See Table–5.111.
- Reverse (1) causes the output value to inversely reflect the digital condition of the input at the R2R resistance network. See Table–5.112.

Table–5.111 Direct Action, Contact-to-Output.

Action [x] = Direct (0)	
R2R Input Contact	**Object Output**
Open	OFF (0.0)

Table–5.111 Direct Action, Contact-to-Output.

Action [x] = Direct (0)	
Closed	ON (100.0)

Table–5.112 Reverse Action, Contact-to-Output.

Action [x] = Reverse (1)	
R2R Input Contact	**Object Output**
Open	ON (100.0)
Closed	OFF (0.0)

Status Flags Output

The Status Flags output is 0 (zero) under normal conditions. This output is set to an enumerated value whenever the DUI Expander algorithm detects an error condition. In addition, Outputs[1] through [5] are all set to a certain state. Errors include resistance under-range, resistance over-range, and improper setup (no valid physical address). Output values for the Status Flags output and Outputs[1] through [5] are shown in Table–5.113 below.

Table–5.113 DUI Expander Status Flag Output Error Codes.

Diagnostic Condition	Status Flag Output	Outputs[1] through [5]
Valid setup and normal object operation.	0	Normal, Current Status
Physical Address set to not active (NA).	1	All not active (NA)
Under-range condition. Total resistance is less than allowed.	2	All ON (100.0)
Over-range condition. Total resistance is greater than allowed.	3	All OFF (0.0)

Note: The Status Flags output can also be used digitally as a value of zero reflects digital OFF and a value greater than zero reflects a digital ON.

Input Resistance Combinations

Table–5.114 and Table–5.115 represent all the possible combinations of the R2R resistor network and the equivalent resistances produced by the network. Table–5.114 reflects the resistances required by MicroNet standard controllers. Table–5.115 reflects the resistances required by the MicroNet MN 800 controller that must include the use of a parallel resistor (Rp).

In general, it is recommended that the DUI module be implemented with 0.1% tolerance resistors to ensure the module's ±3.5 ohm tolerance around each of the nominal resistance switch points. Other tolerances (i.e. 1% resistors) may be used as long as the resulting equivalent resistances of the DUI module fall within the ±3.5 ohm tolerance band.

Table–5.114 and Table–5.115 also provide resistance values that take into account the external wiring, the DUI module tolerance, and the parallel resistance Rp (applicable to the MN 800 only), which is reflected by a ±5 ohm tolerance around each of the nominal resistance switch points. Switch points that are within the specified resistance tolerance bands will guarantee proper DUI operation.

MicroNet Standard Controllers

The external DUI module (including the external wiring) must meet the resistance specifications in Table–5.114, below.

Table–5.114 R2R Resistor Network Equivalent Resistances for MicroNet Standard Controllers.

Input Switch Positions (0 = Switch Open, 1 = Switch Closed)					Standalone DUI Nominal Equivalent Ω without Rp	DUI Module Interface Specifications Nominal Equivalent R ±5 Ω (includes wiring)	
SW1	SW2	SW3	SW4	SW5		Minimum Ω for States Indicated	Maximum Ω for States Indicated
Over-Range Condition					> 1596	1563.0	•
0	0	0	0	0	1557.9	1552.9	1562.9
0	0	0	0	1	1533.0	1528.0	1538.0
0	0	0	1	0	1507.9	1502.9	1512.9
0	0	0	1	1	1483.0	1478.0	1488.0
0	0	1	0	0	1457.9	1452.9	1462.9
0	0	1	0	1	1433.0	1428.0	1438.0
0	0	1	1	0	1407.9	1402.9	1412.9
0	0	1	1	1	1383.0	1378.0	1388.0
0	1	0	0	0	1356.9	1351.9	1361.9
0	1	0	0	1	1332.0	1327.0	1337.0
0	1	0	1	0	1306.9	1301.9	1311.9
0	1	0	1	1	1282.0	1277.0	1287.0
0	1	1	0	0	1256.9	1251.9	1261.9
0	1	1	0	1	1232.0	1227.0	1237.0
0	1	1	1	0	1206.9	1201.9	1211.9
0	1	1	1	1	1182.0	1177.0	1187.0
1	0	0	0	0	1154.9	1149.9	1159.9
1	0	0	0	1	1130.0	1125.0	1135.0
1	0	0	1	0	1104.9	1099.9	1109.9
1	0	0	1	1	1080.0	1075.0	1085.0
1	0	1	0	0	1054.9	1049.9	1059.9
1	0	1	0	1	1030.0	1025.0	1035.0
1	0	1	1	0	1004.9	999.9	1009.9
1	0	1	1	1	980.0	975.0	985.0
1	1	0	0	0	953.9	948.9	958.9
1	1	0	0	1	929.0	924.0	934.0
1	1	0	1	0	903.9	898.9	908.9
1	1	0	1	1	879.0	874.0	884.0
1	1	1	0	0	853.9	848.9	858.9
1	1	1	0	1	829.0	824.0	834.0
1	1	1	1	0	803.9	798.9	808.9
1	1	1	1	1	779.0	774.0	784.0
Under-range Condition					< 741	0	773.9

MicroNet MN 800 Controller

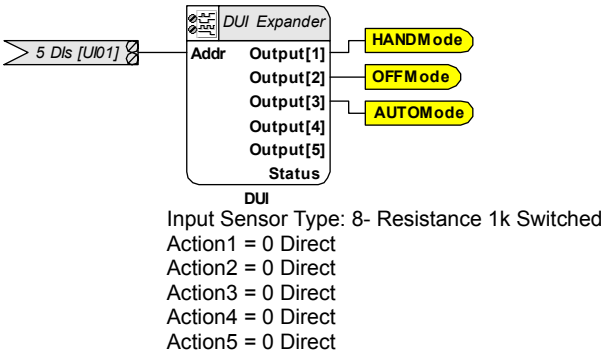
The external DUI module (including the external wiring and parallel resistor, Rp) must meet the resistance specifications shown in Table–5.115, below. The external parallel resistor (Rp=22.1 kohms, 1%) must be placed across the UI input to which the DUI is attached. The DUI module may incorporate this parallel resistor as long as the equivalent resistances from Table–5.115 are maintained.

Table–5.115 R2R Resistor Network Equivalent Resistances for MicroNet MN 800 Controllers.

Input Switch Positions (0 = Switch Open, 1 = Switch Closed)					Standalone DUI Nominal Equivalent Ω with Rp	DUI Module Interface Specifications Nominal Equivalent R ±5 Ω (includes wiring and Rp)	
SW1	SW2	SW3	SW4	SW5		Minimum Ω for States Indicated	Maximum Ω for States Indicated
Over-Range Condition					> 1480	1460.4	•
0	0	0	0	0	1455.3	1450.3	1460.3
0	0	0	0	1	1433.5	1428.5	1438.5
0	0	0	1	0	1411.6	1406.6	1416.6
0	0	0	1	1	1389.7	1384.7	1394.7
0	0	1	0	0	1367.7	1362.7	1372.7
0	0	1	0	1	1345.7	1340.7	1350.7
0	0	1	1	0	1323.6	1318.6	1328.6
0	0	1	1	1	1301.5	1296.5	1306.5
0	1	0	0	0	1278.4	1273.4	1283.4
0	1	0	0	1	1256.2	1251.2	1261.2
0	1	0	1	0	1233.9	1228.9	1238.9
0	1	0	1	1	1211.7	1206.7	1216.7
0	1	1	0	0	1189.3	1184.3	1194.3
0	1	1	0	1	1166.9	1161.9	1171.9
0	1	1	1	0	1144.4	1139.4	1149.4
0	1	1	1	1	1121.9	1116.9	1126.9
1	0	0	0	0	1097.5	1092.5	1102.5
1	0	0	0	1	1075.0	1070.0	1080.0
1	0	0	1	0	1052.3	1047.3	1057.3
1	0	0	1	1	1029.6	1024.6	1034.6
1	0	1	0	0	1006.8	1001.8	1011.8
1	0	1	0	1	984.1	979.1	989.1
1	0	1	1	0	961.2	956.2	966.2
1	0	1	1	1	938.3	933.3	943.3
1	1	0	0	0	914.4	909.4	919.4
1	1	0	0	1	891.5	886.5	896.5
1	1	0	1	0	868.4	863.4	873.4
1	1	0	1	1	845.3	840.3	850.3
1	1	1	0	0	822.1	817.1	827.1
1	1	1	0	1	799.0	794.0	804.0
1	1	1	1	0	775.7	770.7	780.7
1	1	1	1	1	752.4	747.4	757.4
Under-range Condition					< 727	0	747.3

Example

One example use of the DUI Expander object is to monitor the actual status of an H-O-A switch (Hand-Off-Auto). These values could be used for indication and control purposes.



Figure–5.59 DUI Expander Object Used For H-O-A Switch Monitoring.

Use of the Action [n] configuration properties allows inversion of the digital status input when required. The DUI Expander object can also be used to monitor switch panels, for example, hotel occupancy, nurses stations, etc.

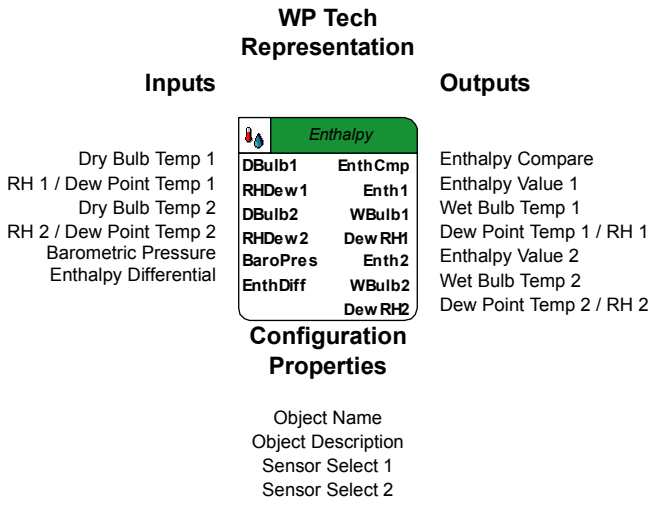
Enthalpy

Object Usage: The Enthalpy object provides a means for calculating the enthalpy (total heat content) of moist air, from one or two air sources. The Enthalpy object algorithm uses either dry bulb temperature and relative humidity, or dew point temperature, to calculate enthalpy, wet bulb temperature, dew point temperature, and / or relative humidity for each source. When two air sources are used, the Enthalpy object compares the enthalpy values of these sources and provides a digital indication of current enthalpy conditions. The object's algorithm also provides barometric pressure compensation for more accurate operation at all elevations.

Device Support:
MN 800 series

Memory Requirements: (per object)
EEPROM: 20 bytes
RAM: 14 bytes

WP Tech Representation



Configuration Properties

Object Name
Object Description
Sensor Select 1
Sensor Select 2

WP Tech Stencil:
Logic and Math Control

Properties

Table–5.116 Enthalpy Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
SenSel1	Sensor Select 1	Class: Analog - Used to determine the type of processing required.	0	Selection configures associated inputs for: Dry bulb temperature and relative humidity (0) Dry bulb temperature and dew point temperature (1)	A value that is negative, out of range, or not active (NA) causes the algorithm to default to "Dry bulb temperature and relative humidity (0)".

Table–5.116 Enthalpy Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
SenSel2	Sensor Select 2	Class: Analog - Used to determine the type of processing required.	0	Selection configures associated inputs for: Dry bulb temperature and relative humidity (0) Dry bulb temperature and dew point temperature (1)	A value that is negative, out of range, or not active (NA) causes the algorithm to default to “Dry bulb temperature and relative humidity (0)”.

Table–5.117 Enthalpy Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
DBulb1	Dry Bulb Temperature 1	Class: Analog - Used by the algorithm to calculate the values of Enthalpy 1, Wet Bulb Temperature 1, and either Dew Point Temperature 1 or Relative Humidity 1, based upon the Sensor Select 1.	—	A value of not active (NA) causes the associated outputs to be set to not active (NA) conditions.
RHDew1	Relative Humidity 1 /Dew Point Temperature 1	Class: Analog - The Relative Humidity 1 / Dew Point Temperature 1 input is configured by the Sensor Select 1 configuration property. When relative humidity is selected, the algorithm interprets the input as a relative humidity percentage value, internally limited to a range of 0.0 to 100.0 percent. Relative Humidity 1 is used to calculate the values of Enthalpy 1, Wet Bulb Temperature 1, and Dew Point Temperature 1. When dew point temperature is selected, the algorithm interprets the input as a dew point temperature value, in either °F or °C, based upon the controller’s “units” selection (English / Metric). The dew point temperature input value is internally limited to a range of 20 to 120 °F (-6.6 to 48.9 °C). Dew Point Temperature 1 is used to calculate the values of Enthalpy 1, Wet Bulb Temperature 1, and Relative Humidity 1.	—	An input value that is below or above the limit range will be held at the minimum or maximum specified limit values accordingly. An input value of not active (NA) causes the associated outputs to be set not active (NA) conditions.
DBulb2	Dry Bulb Temperature 2	Class: Analog - Used by the algorithm to calculate the values of Enthalpy 2, Wet Bulb Temperature 2, and either Dew Point Temperature 2 or Relative Humidity 2, based upon the Sensor Selection 2.	—	A value of not active (NA) causes the associated outputs to be set to not active (NA) conditions.

Table–5.117 Enthalpy Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
RHDew2	Relative Humidity 2 /Dew Point Temperature 2	Class: Analog - The Relative Humidity 2 / Dew Point Temperature 2 input is configured by the Sensor Select 2 configuration property. When relative humidity is selected, the algorithm interprets the input as a relative humidity percentage value, internally limited to a range of 0.0 to 100.0 percent. Relative Humidity 2 is used to calculate the values of Enthalpy 2, Wet Bulb Temperature 2, and Dew Point Temperature 2. When dew point temperature is selected, the algorithm interprets the input as a dew point temperature value, in either °F or °C, based upon the controller's "units" selection (English / Metric). The dew point temperature input value is internally limited to a range of 20 to 120 °F (-6.6 to 48.9 °C). Dew Point Temperature 2 is used to calculate the values of Enthalpy 2, Wet Bulb Temperature 2, and Relative Humidity 2.	—	An input value that is below or above the limit range will be held at the minimum or maximum specified limit values accordingly. An input value of not active (NA) causes the associated outputs to be set not active (NA) conditions.
BaroPres	Barometric Pressure	Class: Analog - Normally connected to a value that represents the current atmospheric pressure conditions. The algorithm interprets the value in either in. Hg or mm Hg, based upon the controller's "units" selection (English / Metric). This input value is internally limited to a range of 19.94 to 39.79 in. Hg (506.5 to 1010.7 mm Hg).	—	An input value that is below or above the limit range will be held at the minimum or maximum specified limit values accordingly. A Barometric Pressure of not active (NA) causes the value to default to a barometric pressure of 29.92 in. Hg. (760 mm. Hg.).
EnthDiff	Enthalpy Differential	Class: Analog - Defines the differential used to calculate the Digital ON / OFF control points for the Enthalpy Compare output. The algorithm interprets the value in either Btu per pound or kilojoules per kilogram, based upon the controller's "units" selection (English / Metric).	—	An input value that is negative or not active (NA) causes the algorithm to default to an Enthalpy Differential value of zero

Table–5.118 Enthalpy Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
EnthCmp	Enthalpy Compare	Class: Digital - Provides a digital indication of current enthalpy conditions. The algorithm compares the Enthalpy 1 value to the Enthalpy 2 value and applies the Enthalpy Differential for determining Digital ON / OFF control. Under normal operation, no change to the Enthalpy Compare output is made if the comparison result is within the calculated differential range. Upon reset, or return from Enthalpy 1 or Enthalpy 2 not active conditions, the algorithm performs the trip point comparison and sets the Enthalpy Compare output accordingly. However, Enthalpy 1 and Enthalpy 2 values that are found to be within the calculated differential range will cause the algorithm to set the Enthalpy Compare output to Digital OFF.	A not active (NA) Enthalpy 1 or Enthalpy 2 value will cause the object to set the Enthalpy Compare output to a not active (NA) value.
Enth1	Enthalpy Value 1	Class: Analog - Represented by the symbol “h”, Enthalpy refers to the heat content of the moist air. It is expressed in either Btu per pound or kilojoules per kilogram of dry air, based upon the controller’s “units” selection (English / Metric).	Value is set to a not active (NA) whenever Dry Bulb Temperature 1 and either Dew Point Temperature 1 or Relative Humidity 1 inputs (based upon the Sensor Select 1) are not active (NA).
WBulb1	Wet Bulb Temperature 1	Class: Analog - Reflects the calculated wet bulb temperature, in either °F or °C, based upon the controller’s “units” selection (English / Metric).	Value is set to a not active (NA) whenever Dry Bulb Temperature 1 and either Dew Point Temperature 1 or Relative Humidity 1 inputs (based upon the Sensor Select 1) are not active (NA).
DewRH1	Dew Point Temperature 1 Relative Humidity 1	Class: Analog - This output value is based upon the Sensor Select 1 configuration property. Selecting dry bulb temperature / relative humidity causes the algorithm to calculate the output as a dew point temperature, reflecting the calculated dew point temperature in either °F or °C, based upon the controller’s “units” selection (English / Metric). Selecting dry bulb temperature / dew point temperature causes the algorithm to calculate the output as relative humidity, reflecting the relative humidity percentage from 0.0 to 100.0%.	Value is set to a not active (NA) whenever Dry Bulb Temperature 1 and either Dew Point Temperature 1 or Relative Humidity 1 inputs (based upon the Sensor Select 1) are not active (NA).
Enth2	Enthalpy Value 2	Class: Analog - Represented by the symbol “h”, Enthalpy refers to the heat content of the moist air. It is expressed in either Btu per pound or kilojoules per kilogram of dry air, based upon the controller’s “units” selection (English / Metric).	Value is set to a not active (NA) whenever Dry Bulb Temperature 2 and either Dew Point Temperature 2 or Relative Humidity 2 inputs (based upon the Sensor Select 2) are not active (NA).

Table–5.118 Enthalpy Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
WBulb2	Wet Bulb Temperature 2	Class: Analog - Reflects the calculated wet bulb temperature, in either °F or °C, based upon the controller's "units" selection (English / Metric).	Value is set to a not active (NA) whenever Dry Bulb Temperature 2 and either Dew Point Temperature 2 or Relative Humidity 2 inputs (based upon the Sensor Select 2) are not active (NA).
DewRH2	Dew Point Temperature 2 Relative Humidity 2	Class: Analog - This output value is based upon the Sensor Select 2 configuration property. Selecting dry bulb temperature / relative humidity causes the algorithm to calculate the output as a dew point temperature, reflecting the calculated dew point temperature in either °F or °C, based upon the controller's "units" selection (English / Metric). Selecting dry bulb temperature / dew point temperature causes the algorithm to calculate the output as relative humidity, reflecting the relative humidity percentage from 0.0 to 100.0%.	Value is set to a not active (NA) whenever Dry Bulb Temperature 2 and either Dew Point Temperature 2 or Relative Humidity 2 inputs (based upon the Sensor Select 2) are not active (NA).

Applying the Object

The Enthalpy object provides a means for calculating the enthalpy (total heat content) of moist air from one or two air sources. The object's algorithm uses dry bulb temperature and either relative humidity or dew point temperature to calculate enthalpy, wet bulb temperature, dew point temperature, and / or relative humidity for each source. When two air sources are used, the object compares their enthalpy values and provides a digital indication of current enthalpy conditions. The object's algorithm also provides barometric pressure compensation, for more accurate operation at all elevations.

Enthalpy, represented by the symbol "h", refers to the heat content of the moist air. Enthalpy is expressed in either Btu per pound or kilojoules per kilogram of dry air, based upon the controller's "units" selection (English / Metric).

In a typical air handler, the Enthalpy object is used to determine the enthalpy, or total heat content, of both outside air and return air sources. The control algorithm can be configured so that, when in a cooling mode, it uses the air stream with the least enthalpy, to minimize overall cooling costs.

Configuration

The Enthalpy object includes a Sensor Select configuration property for each pair of sensor inputs. The algorithm uses Sensor Select 1 and Sensor Select 2 to determine the processing required. A Sensor Select value of 0 (zero) configures inputs for dry bulb temperature and relative humidity, and causes the algorithm to calculate associated outputs of enthalpy value, wet bulb temperature, and dew point temperature. A Sensor Select value of 1 configures inputs for dry bulb temperature and dew point temperature, and causes the algorithm to calculate associated outputs of enthalpy value, wet bulb temperature, and relative humidity. A Sensor Select value that is negative, out of range, or not active (NA) causes the algorithm to default to a Sensor Select value of zero.

Inputs

Dry Bulb Temperature

The Dry Bulb Temperature input is normally connected to a value that represents the dry bulb temperature. The algorithm interprets the value in either °F or °C, based upon the controller's "units" selection (English or Metric). The Dry Bulb Temperature 1 and Dry Bulb Temperature 2 values are used by the algorithm to calculate the corresponding Enthalpy, Wet Bulb Temperature, and either Dew Point Temperature or Relative Humidity values, based upon the associated Sensor Select value.

The dry bulb temperature input value is internally limited to a range of 32 to 120 °F (0 to 48.9 °C). An input value that is below or above the limit range will be held at the minimum or maximum specified limit values accordingly. A Dry Bulb Temperature of not active (NA) causes the associated outputs to be set to not active (NA) conditions.

Relative Humidity / Dew Point Temperature

The Relative Humidity / Dew Point Temperature input is configured by the Sensor Select configuration property.

When relative humidity is selected, the algorithm interprets the input as a relative humidity percentage value, internally limited to a range of 1.0 to 100.0%. Relative Humidity 1 and Relative Humidity 2 are used to calculate the corresponding Enthalpy, Wet Bulb Temperature, and Dew Point Temperature values. An input value that is below or above the limit range will be held at the minimum or maximum specified limit values accordingly. A value of not active (NA) causes the associated outputs to be set to not active (NA) conditions.

When dew point temperature is selected, the algorithm interprets the input as a dew point temperature value, in either °F or °C, based upon the controller's "units" selection (English / Metric). The dew point temperature input value is internally limited to a range of 20 to 120 °F (-6.6 to 48.9 °C). An input value that is below or above the limit range will be held at the minimum or maximum specified limit values, accordingly.

Dew Point Temperature 1 and Dew Point Temperature 2 are used to calculate the corresponding Enthalpy, Wet Bulb Temperature, and Relative Humidity values. A Dew Point Temperature value of not active (NA) causes the associated outputs to be set to not active (NA) conditions.

Barometric Pressure

The Barometric Pressure input is normally connected to a value that represents the current atmospheric pressure conditions. The algorithm interprets the value in either in. Hg or mm Hg, based upon the controller's "units" selection (English / Metric). Variations in atmospheric pressure (barometric pressure), due to elevation above or below sea level, can significantly affect the values of the various calculated properties. The algorithm uses the barometric pressure value for atmospheric compensation of all sensor inputs within the object. Barometric pressure compensation is typically required for installations at altitudes greater than 2000 feet (600 meters).

The Barometric Pressure input value is internally limited to a range of 19.94 to 39.79 in. Hg (506.5 to 1010.7 mm Hg). An input value that is below or above the limit range will be held at the minimum or maximum specified limit values, accordingly. A Barometric Pressure of not active (NA) causes the value to default to a barometric pressure of 29.92 in. Hg (760 mm Hg).

Enthalpy Differential

The Enthalpy Differential input value defines the differential used to calculate the Digital ON / OFF control points for the Enthalpy Compare output. The algorithm interprets the value in either Btu per pound or kilojoules per kilogram, based upon the controller's "units" selection (English / Metric). An Enthalpy Differential value that is negative or not active (NA) causes the algorithm to default to an Enthalpy Differential value of 0 (zero).

Outputs

Enthalpy Compare

The Enthalpy Compare output provides a digital indication of current enthalpy conditions. The algorithm compares the Enthalpy 1 value to the Enthalpy 2 value and applies the Enthalpy Differential to determine Digital ON / OFF control. The calculated Enthalpy Compare output trip points are based upon the following equations.

When Enthalpy 1 ≤ Enthalpy 2: The algorithm will test for the Digital OFF enthalpy trip point. The Enthalpy Compare output is set to Digital OFF when Enthalpy 1 ≤ (Enthalpy 2 – Enthalpy Differential). The minimum value of Enthalpy 2 – Enthalpy Differential is limited to 0 (zero) (negative result values are not allowed).

When Enthalpy 1 > Enthalpy 2: The algorithm will test for the Digital ON enthalpy trip point. The Enthalpy Compare output is set to Digital ON when Enthalpy 1 > (Enthalpy 2 + Enthalpy Differential).

No change in the Enthalpy Compare output will be made if the comparison result is within the calculated differential range, that is, neither above nor below the calculated enthalpy output trip points. A not active (NA) Enthalpy 1 or Enthalpy 2 value will cause the object to set the Enthalpy Compare output to a not active (NA) value.

Upon reset or return from Enthalpy 1 or Enthalpy 2 not active conditions, the algorithm performs the trip point comparison and sets the Enthalpy Compare output accordingly. However, Enthalpy 1 and Enthalpy 2 values that are found to be within the calculated differential range will cause the algorithm to set the Enthalpy Compare output to Digital OFF.

Enthalpy 1 and Enthalpy 2 represent enthalpy values. Enthalpy, designated by the symbol "h", refers to the heat content of moist air, and is expressed in either Btu per pound or kilojoules per kilogram of dry air, based upon the controller's "units" selection (English / Metric).

The Enthalpy 1 or Enthalpy 2 value is set to a not active (NA) whenever the corresponding Dry Bulb Temperature and either Dew Point Temperature or Relative Humidity (based upon the associated Sensor Select value) are not active (NA).

Wet Bulb Temperature

Wet Bulb Temperature 1 and Wet Bulb Temperature 2 reflect the calculated wet bulb temperature in, either *°F or °C*, based upon the controller's "units" selection (English / Metric). The process of evaporation has a cooling effect that is directly related to the moisture content of the air. The wet bulb temperature is the temperature at which water (liquid or solid), by evaporating into moist air at a given dry bulb temperature and humidity ratio, can bring the air to its saturation point adiabatically, while a constant barometric pressure is maintained. Wet bulb temperatures are always lower than dry bulb temperatures, unless the air is completely saturated (100% relative humidity).

The Wet Bulb Temperature 1 or Wet Bulb Temperature 2 value is set to a not active (NA) whenever the corresponding Dry Bulb Temperature and either Dew Point Temperature or Relative Humidity (based upon the associated Sensor Select value) are not active (NA).

Dew Point Temperature / Relative Humidity

The Dew Point Temperature / Relative Humidity output value is based upon the Sensor Select configuration property.

When dry bulb temperature / relative humidity is selected, the algorithm calculates the output (Dew Point Temperature 1 or Dew Point Temperature 2) as a dew point temperature value, in either °F or °C, based upon the controller's "units" selection (English / Metric). Dew point temperature is the temperature to which a given sample of air must be cooled for moisture condensation to occur. For moisture saturated air, in which relative humidity is at 100.0%, the dry bulb, wet bulb, and dew point temperatures are all equal. The Dew Point Temperature value is set to a not active (NA) whenever the corresponding Dry Bulb Temperature and either Dew Point Temperature or Relative Humidity (based upon the associated Sensor Select value) are not active (NA).

When dry bulb temperature / dew point temperature is selected, the algorithm calculates the output (Relative Humidity 1 or Relative Humidity 2) as a relative humidity percentage value, limited to a range of 0.0 to 100.0%. Relative humidity expresses the relationship of the amount of moisture in the air to the amount the air would hold if saturated at the dry bulb temperature. The Relative Humidity output value is set to a not active (NA) whenever the corresponding Dry Bulb Temperature and either Dew Point Temperature or Relative Humidity (based upon the associated Sensor Select value) are not active (NA).

Input / Output Units

The input and output units are based upon the (English / Metric) units selection made during the controller setup procedure. The following table provide a quick reference for these units.

Table–5.119 English and Metric Units Used with Controller Inputs and Outputs.

	Controller Units Selection	
	English	**Metric**
Inputs		
Dry Bulb Temperature 1	°F	°C
Relative Humidity 1 or Dew Point Temperature 1	% °F	% °C
Dry Bulb Temperature 2	°F	°C
Relative Humidity 2 or Dew Point Temperature 2	% °F	% °C
Barometric Pressure	in. Hg	mm. Hg
Enthalpy Differential	Btu / lb	kJ / kg
Outputs		
Enthalpy Compare	—	—
Enthalpy 1	Btu / lb	kJ / kg
Wet Bulb Temperature 1	°F	°C
Dew Point Temperature 1 or Relative Humidity 1	°F %	°C %
Enthalpy 2	Btu / lb	kJ / kg
Wet Bulb Temperature 2	°F	°C
Dew Point Temperature 2 or Relative Humidity 2	°F %	°C %

Standard Atmospheric Data for Altitudes

Variations in atmospheric pressure (barometric pressure), due to elevation above or below sea level, can significantly affect the values of the various calculated properties. The temperature and barometric pressure of ambient air vary considerably with altitude, as well as with local geographic and weather conditions. The standard atmosphere, which establishes atmospheric properties at sea level under standard conditions, provides a reference for estimating properties at other altitudes. The barometric pressure at sea level, under standard conditions, is 29.921 in. Hg (760 mm Hg). The Enthalpy object's algorithm uses the standard barometric pressure to calculate the atmospheric compensation of all sensor inputs within the object. Barometric pressure compensation is typically required for installations at altitudes greater than 2000 feet (600 meters). Table–5.120 provides a quick reference for barometric pressures at various altitudes, under standard conditions.

Table–5.120 Barometric Pressures Under Standard Conditions.

English Units		Metric Units	
Altitude (ft)	Barometric Pressure (in. Hg)	Altitude (m)	Barometric Pressure (mm Hg)
-1000	31.02	-1000	854
-500	30.47	-500	806
0	29.921	0	760
500	29.38	250	737
1000	28.86	500	716
2000	27.82	750	695
3000	26.82	1000	674
4000	25.82	1250	654
5000	24.90	1500	634
6000	23.98	1750	615
7000	23.09	2000	596
8000	22.22	2500	560
9000	21.39	3000	526
10000	20.58	3500	493

Conversions

Table–5.121 provides the English and metric conversions employed in the Enthalpy object.

Table–5.121 English and Metric Conversions Used in the Enthalpy Object.

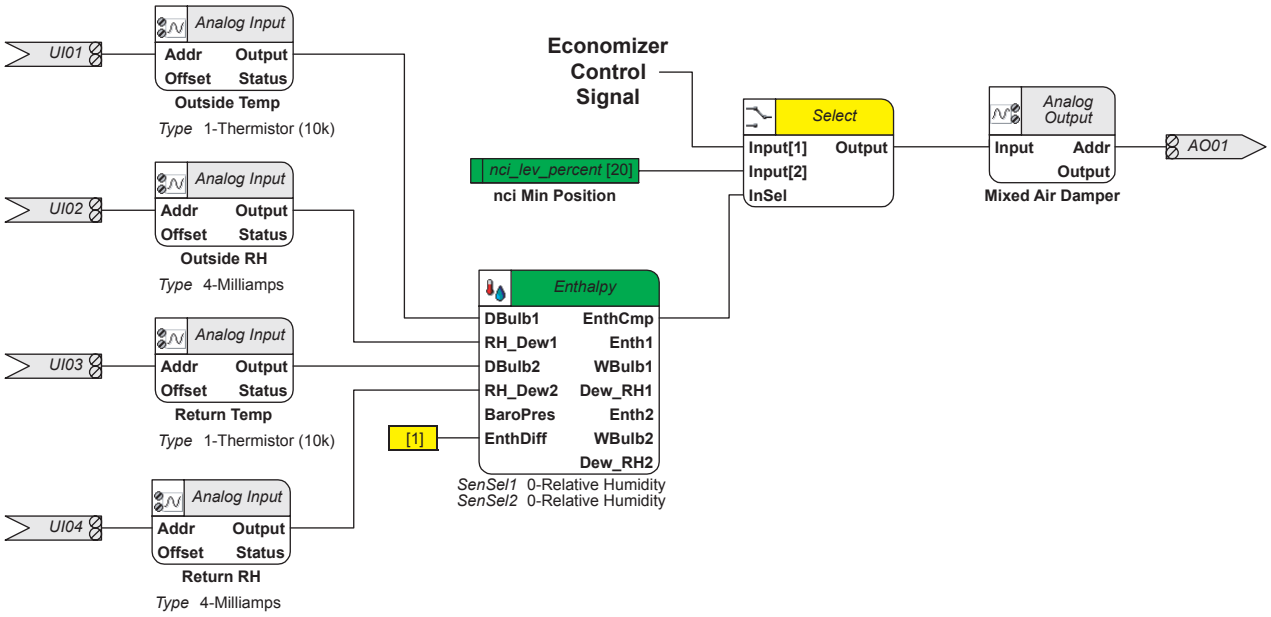
Atmospheric Property	English to Metric		Metric to English	
	Conversion	Factor	Conversion	Factor
Enthalpy	Btu/lb to kJ/kg	(Btu/lb x 2.326) - 17.88	kJ/kg to Btu/lb	(kJ/kg + 17.88) x 0.4299
Pressure	in. Hg to mm Hg	in. Hg x 25.4	mm Hg to in. Hg	mm Hg x 0.03937
	psia to mm Hg	psia x 51.715	mm Hg to psia	mm Hg x 0.017327
Temperature	°F to °C	(°F - 32) ÷ 1.8	°C to °F	(°C x 1.8) + 32

Example

The following example discusses the use of an Enthalpy object to provide a more economical means of controlling the use of outside air:

Economizer cycles that are based upon dry bulb temperature alone are not always the most economical means for controlling mixed air. Such is often the case in very humid climates, where the total heat (enthalpy) of the outside air may be greater than that of the return air, even though its dry bulb temperature is lower. Since it is the *total* heat that the cooling coil must remove from the air to maintain the desired condition, it is more economical in this case to hold outside air to a minimum.

In the above situation, the Enthalpy object would be used for enthalpy optimization with mixed air control. To accomplish this, two air sources are used, outside air and return air. The Enthalpy object would be applied in such a way that outside air is selected when its enthalpy is less than the return air enthalpy. When conditions are reversed, so that the outside air enthalpy is greater than the return air enthalpy, the outside air dampers would be set to an adjustable minimum position.



Figure–5.60 Enthalpy Object Used for Enthalpy Optimization with Mixed Air Control.

Event Indicator

Object Usage: The Event Indicator object is a special purpose point-type object with a physical hardware address for a digital output (DO) point. It operates like a direct Binary Output object, but can also generate an interrupting pulse (flick) used to signal an approaching event. Typical use is for a lighting load to warn an occupant before actually turning OFF the lights. The Input to the object is an analog value representing the known time remaining before a digital transition, typically supplied by a Control Override object or a OvrdTimeRemain sensor tag (MicroNet sensor). The Event Indicator object compares this input to the assigned Event Time and when matching, cycles (flicks) the Output for the time specified in the Event Duration.

Device Support: (all firmware revisions)

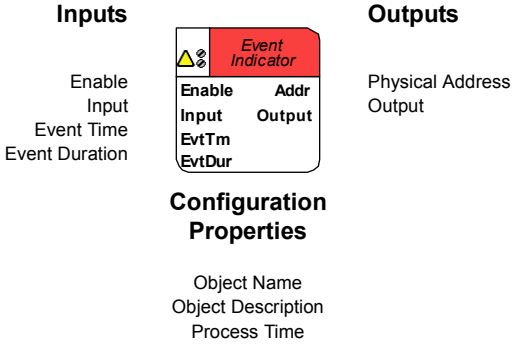
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 14 bytes
RAM: 20 bytes (standard controllers)
6 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Timer and Sequence Control

Properties

Table–5.122 Event Indicator Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.123 Event Indicator Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Enable	Enable	Class: Digital - An ON or not active (NA) permits an event indication sequence (flick) to operate. An OFF disables the event indication sequence (flick), and the Output directly tracks the Input.	—	
Input	Input	Class: Analog - An input value greater than 0 (zero) turns the Output ON (with the event indicator sequence followed).	-163.83 to 16383	An input value of zero, negative, or not active (NA) turns the Output OFF.
EvtTm	Event Time	Class: Analog - The value compared to the Input for an event indication sequence (flick).	-163.83 to 16383 minutes	A value of 0 (zero), a negative value, or not active (NA) at this input results in no event indication.
EvtDur	Event Duration	Class: Analog - The length of time (in seconds) for the event indication (flick). ***Note:*** A not active (NA) may result in short-cycling of the outputs.	0.0 to 10,000 seconds	If negative, 0 (zero), or not active (NA), it disables the event indication sequence. Values greater than 10,000 will time out at 10,000 seconds.

Table–5.124 Event Indicator Object Output Properties.

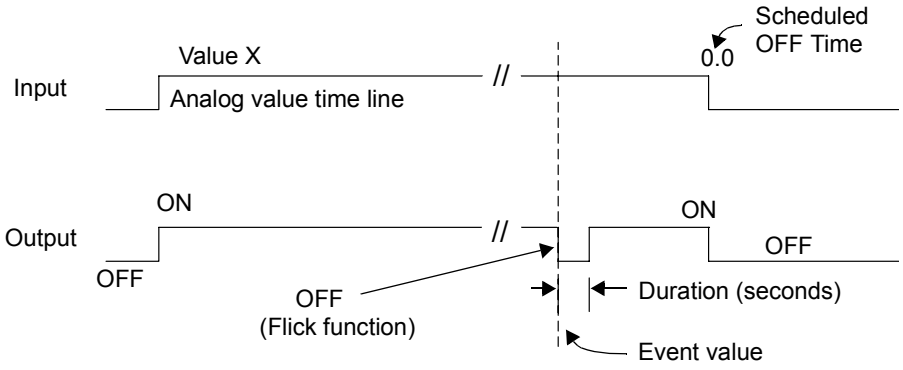
Abbrev.	Name	Class / Description	Valid Values
Addr	Physical Address	Class: Analog - Defines the physical hardware address (output terminal point on the controller) assigned to the Event Indicator object.	Dependent on the controller platform selected.
Output	Output	Class: Digital - This output indicates the calculated digital output state. The output is ON whenever the Input value is greater than 0 (zero) and an event indication sequence (flick) is not in progress. The output is OFF during an event indication sequence (flick), or whenever the Input value is zero, negative, or not active (NA).	OFF (0.0) ON (100.0)

Applying the Object

The Event Indicator object is typically used in place of a Binary Output object to control a lighting load with a time-of-day schedule or a user override. In this application, the Event Indicator object provides a lighting “flick” function to warn occupants that the lighting load will soon be OFF.

The occurrence of the flick is determined by the analog value received at the Input, typically from a Schedule or Control Override object. This value counts down to zero during an approaching ON-to-OFF transition. When the Input matches the value at the Event Time input (in minutes), an event sequence transition (flick) occurs. A flick is an OFF pulse (in seconds) at the controller hardware (DO) and object Output, equal in duration to the value at the Event Duration input. Following the flick, the remaining scheduled ON time continues (as determined by the Input value’s remaining countdown time to zero).

Figure–5.61 below illustrates the operation of the Event Indicator object.

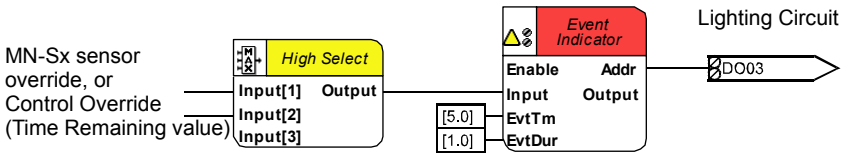


Figure–5.61 Timing Diagram for an Event Indicator Object.

Example

The Event Indicator object in Figure–5.62 is used to control a lighting load and provide a flick alert before a MN sensor-initiated override expires or a Control Override object returns to OFF. The lights will “flick” OFF (for one second) five minutes before the scheduled OFF time.

In this example, Event Time is 5.0 and Event Duration is 1.0.



Figure–5.62 Example Event Indicator Object in a Lighting Application.

Note: A not active (NA) at the Event Duration input (EvtDur) may result in short-cycling of the outputs.

EXOR

Object Usage: The EXOR object (Exclusive OR) is a two-input logic object for use with OFF and ON digital values (DV). The output of the object is a digital ON whenever the inputs are both valid and *not* the same. A digital OFF results whenever the inputs are both valid and are the same. A not active (NA) or unconnected input is ignored.

Output = Exclusive OR (In1, In2)

Device Support: (all firmware revisions)

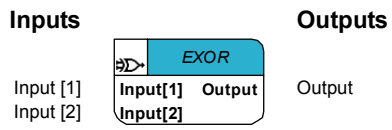
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 8 bytes
RAM: 10 bytes (standard controllers)
2 bytes (MN 800)

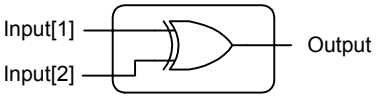
WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

Logic



EXOR

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Digital Logic Objects

Object Name	Digital Object Algorithm (all are three-input unless noted)
AND / AND	In1 AND In2 AND In3
AND / OR	(In1 AND In2) OR In3
Clocked SR	Clocked Set-Reset Flip-Flop Logic
EXOR	Two-input, Exclusive OR
Latch	Digital Sample and Hold or Latch
OR / AND	(In1 OR In2) AND In3
OR / OR	In1 OR In2 OR In3
SR Flip-Flop	Two-input, Set-Reset Flip-Flop Logic

Properties

Table–5.125 EXOR Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.126 EXOR Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Digital - The first input evaluated for a digital ON or OFF. A not active (NA) is ignored. **(In1** XOR In2)	—	See the Truth Table for all possible input combinations and corresponding outputs.
Input[2]	Input [2]	Class: Analog - The second input evaluated for a digital ON or OFF. A not active (NA) is ignored. (In1 XOR **In2**)	—	See the Truth Table for all possible input combinations and corresponding outputs.

Table–5.127 EXOR Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The output indicates the result of the Exclusive OR logic algorithm, meaning: • The output is ON if Input[1] and Input[2] are valid and *not* the same. • The output is OFF if Input[1] and Input[2] are valid and *are* the same. • If not active (NA) is present both inputs, the output is set to NA.	OFF (0.0) ON (100.0)

Applying the Object

The EXOR object provides an Exclusive OR logic function for two digital inputs. The object output is a digital ON whenever the digital states of the two inputs are opposite (not the same). Not active (NA) inputs are ignored unless both inputs are NA, in which case the Output is NA. Table–5.128 shows all possible input to output combinations.

Table–5.128 Truth Table for EXOR Object.

Input[1]	Input[2]	Output
OFF	OFF	OFF
OFF	ON	ON
OFF	NA	OFF
ON	OFF	ON
ON	ON	OFF
ON	NA	ON
NA	OFF	OFF
NA	ON	ON
NA	NA	NA

As with other logic objects, Inputs to this object are typically digital values from a connection to another object's output, represented numerically with a 0 (OFF) or 100 (ON). However, Inputs can also process analog values or a value from an assigned constant. An Input with an analog value or constant is evaluated as OFF if the value is equal to or less than zero (0.0), or ON if the Input has any positive value greater than zero.

Fan Speed

Object Usage: The Fan Speed object can be configured as a point-type object for sequenced control of up to three hardware digital outputs (DOs) to support a one-, two-, or three-speed fan, or it can be configured to provide a proportional analog signal for control of a variable speed fan. The object provides special fan output sequences including a minimum on time for digital outputs, a purge (shutdown) time, an analog start point, and a “kick start” sequence for starting fractional horsepower motors. Output states of the Fan Speed object are determined by the value received on the Required Speed input, along with other property settings.

Device Support: (all firmware revisions)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4

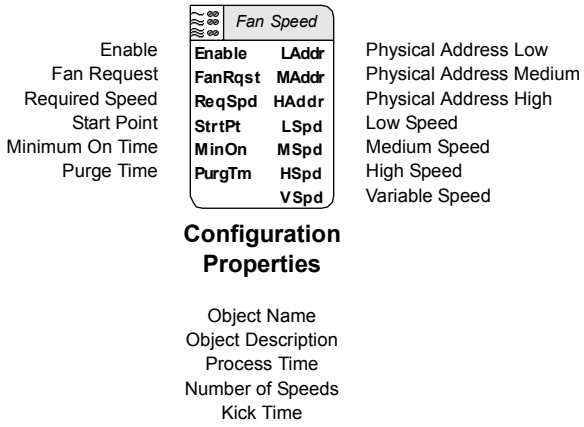
MNL-11Rxx, -13Rxx
where xx = F2 or F3

MN 800 series

Memory Requirements: (per object)

EEPROM: 26 bytes
RAM: 42 bytes (standard controllers)
16 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Number of Speeds
Kick Time

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.129 Fan Speed Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.129 Fan Speed Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
NumSpd	Number of Speeds	Class: Analog - Defines the controlled fan type, type of control, and the object Output(s) used. • Single Speed (digital) uses the High Speed and Variable Speed outputs. • Two Speed (digital) uses the Low, High, and Variable Speed outputs. • Three Speed (digital) uses the Low, Medium, High, and Variable Speed outputs. • Analog (analog) uses only the Variable Speed output.	0 - Single Speed	0 - Single Speed 1 - Two Speed 2 - Three Speed 3 - Analog	Analog selection produces a proportional value output
KickTm	Kick Time	Class: Analog - A value greater than 1.0 enables a kick start / kick time sequence (in seconds). This value defines the time the kick is in effect, which executes when a fan startup sequence is initiated. Value of 0 disables the kick feature.	0	0, 1.0 to 10,000 seconds	A Kick sequence is High Speed output ON (if digital control) or Variable Speed output of 100% (if analog control).

Table–5.130 Fan Speed Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Enable	Enable	Class: Digital - A not active (NA) or ON enables the Fan Speed function. An OFF disables the Fan Speed function, setting all digital outputs OFF and the Variable Speed output to 0.0%. All timers are reset.	—	This input has the highest priority of all inputs.
FanRqst	Fan Requested	Class: Digital - An ON allows the outputs to be set to the speed configuration determined by the object algorithm using the Required Speed / Start Point values. An OFF or not active (NA) sets all digital outputs OFF and the Variable Speed output to 0.0% (upon completion of any remaining timeouts).	—	
ReqSpd	Required Speed	Class: Analog - The actual speed used by the object algorithm to determine the output configuration. A not active (NA) or negative value sets all digital outputs to OFF and the Variable Speed output to 0.0% upon completion of any remaining timeouts.	0.0 to 100.0%	See the Fan Speed Chart.
StrtPt	Start Point	Class: Analog - Valid if the Number of Speeds is set to Analog. Represents the minimum value at the Variable Speed output when executing a run sequence.	0.0 to 100%	A negative or not active (NA) is evaluated as equal to 0.0%.
MinOn	Minimum On Time	Class: Analog - The minimum time (in seconds) that a digital output must remain ON once that particular output or speed has been activated to ON. This timeout does not operate until completion of a kick sequence (if applicable). Not active (NA) is equivalent to 0.0.	0.0 to 10,000 (seconds)	Prevents short-cycling of output speeds when the Speed Required input is fluctuating, and is applied between speeds regardless of the size of the change.

Table–5.130 Fan Speed Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
PurgTm	Purge Time	Class: Analog - The amount of time (in seconds) that an output must remain active after the algorithm determines all outputs should be set to OFF or 0.0%.	0.0 to 10,000 (seconds)	A negative or not active (NA) is evaluated as equal to 0.0%.

Table–5.131 Fan Speed Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
LAddr	Physical Address Low	Class: Analog - Defines the physical hardware address (digital output terminal point on the controller) assigned to the Fan Speed object low speed function. Used in Two Speed and Three Speed configurations.	Dependent on the controller platform selected.
MAddr	Physical Address Medium	Class: Analog - Defines the physical hardware address (digital output terminal point on the controller) assigned to the Fan Speed object medium speed function. Used only in Three Speed configurations.	Dependent on the controller platform selected.
HAddr	Physical Address High	Class: Analog - Defines the physical hardware address (digital output terminal point on the controller) assigned to the Fan Speed object high speed function. Used in Single Speed, Two Speed, and Three Speed configurations.	Dependent on the controller platform selected.
LSpd	Low Speed	Class: Digital - Toggles from OFF to ON to indicate low speed operation. This output is active only when the Number of Speeds is Two Speed or Three Speed. For other configurations, the output is held to OFF.	OFF (0.0) ON (100.0)
MSpd	Medium Speed	Class: Digital - Toggles from OFF to ON to indicate medium speed operation. This output is active only when the Number of Speeds is Three Speed. For other configurations, the output is held to OFF.	OFF (0.0) ON (100.0)
HSpd	High Speed	Class: Digital - Toggles from OFF to ON to indicate high speed operation. This output is active for all digital configurations (Single Speed, Two Speed or Three Speed). In Analog configuration, the output is held OFF.	OFF (0.0) ON (100.0)
VSpd	Variable Speed	Class: Analog - The calculated speed value dependent upon the Number of Speeds configuration and the Required Speed input value. If the Number of Speeds is Analog, the Start Point value determines the lowest value in a run sequence. See the speed chart for further details.	0 to 100%

Applying the Object

The Fan Speed object can be used for control of a multiple speed fan or a variable speed fan. For a multiple speed fan, the object acts as a digital point-type object, able to sequence from one to three hardware digital outputs (in place of one or more Binary Output objects). For a variable speed fan, the object produces an analog output value that is the calculated variable speed. An additional Analog Output object is required to actually drive the hardware analog output.

Number of Speeds Property

The Fan Speed object is configured for control of either one of the digital fan types (Single Speed, Two Speed, or Three Speed) or an Analog (variable) speed fan with the Number of Speeds configuration property. The outputs of the Fan Speed object behave differently with each configuration type (in relation to the Required Speed input), as shown in the following chart Figure-5.132.

Table–5.132 Fan Speed Chart.

Configuration	Input	Outputs			
Number of Speeds	**(Required Speed[a])**	**Low Speed**	**Medium Speed**	**High Speed**	**Variable Speed**
Single	0.0%	OFF	OFF	OFF	0.0%
	>0.0% to <100.0%	OFF	OFF	ON	100.0%
Two	0.0%	OFF	OFF	OFF	0.0%
	>0.0% to <50.0%	ON	OFF	OFF	50.0%
	>50.0% to <100.0%	OFF	OFF	ON	100.0%
Three	0.0%	OFF	OFF	OFF	0.0%
	>0.0% to <33.0%	ON	OFF	OFF	33.3%
	>33.0% to <66.0%	OFF	ON	OFF	66.0%
	>66.0% to <100.0%	OFF	OFF	ON	100.0%
Analog	0.0%	OFF	OFF	OFF	0.0%
	>0.0% to <100.0%	OFF	OFF	OFF	Start Point value to 100%

a. When the Required Speed input fluctuates near a speed change threshold, the MinOn input function can provide short-cycle protection.

In any speed configuration, the Fan Speed object provides a “Kick sequence” for starting a fan from an OFF state. The Kick sequence is helpful in starting fractional horsepower fans by providing a full output (High Speed ON or 100%) for a configurable time period before its timeout allows the Required Speed input to be followed.

Other fan control features are unique to the two configuration types (digital or analog), each is described separately ahead.

Enable / Disable

The highest priority input of the Fan Speed object is the Enable input. A not active (NA) or ON at this input enables the Fan Speed object control sequences. If an OFF is present at the Enable input, all digital outputs are immediately set to OFF, the Variable Speed output is held at 0.0%, and all Fan Speed object timers (minimum on, kick sequence) are reset to zero.

In addition, the Fan Request input requires an ON for the outputs to become active. A not active (NA) or OFF at the Fan Request input sets all digital outputs to OFF and the Variable Speed output to 0.0%, but only after any Minimum On Time and Purge Time periods have expired.

Digital Fan Speed Control (Single, Two, or Three Speed)

The Fan Speed object must be enabled (Enable set to NA or ON). A Fan Request of ON causes the Fan Speed object to use the Required Speed input value to determine the proper output response.

A fan run sequence begins when the Required Speed rises above 0.0%. Outputs activate differently depending on Kick sequence configuration.

Run Sequence (no Kick Time)

With no assigned Kick sequence (Kick Time = 0, the default), the outputs go to the state corresponding to the value at the Required Speed input for the particular speed configuration. One of the digital speed Outputs will be ON and the Variable Speed output will have a corresponding set value, as shown in Table–5.132.

Run Sequence (with Kick Time)

If a Kick start sequence is assigned (Kick Time > 0), the High Speed output goes ON for the number of seconds defined in Kick Time. The Variable Speed output is also 100% during this Kick sequence. When this Kick timer expires, the outputs go to the state corresponding to the value at the Required Speed input for the particular speed configuration. One of the digital speed Outputs will be ON and the Variable Speed output will have a corresponding set value. See Table–5.132.

Speed Change and Minimum On Time

Any assigned kick sequence must first be completed before the required speed input is evaluated. Upon kick sequence completion, the requested speed will be placed at the outputs. The digital outputs as well as the variable speed output will be configured based upon the percent value applied.

At this point, a timeout defined by the value assigned to the minimum on time input is processed before an output speed change is allowed. The Minimum On Time (in seconds) is applied between speeds regardless of the magnitude or direction of change. No further output changes are allowed until the Minimum On timer expires, including an OFF at Fan Request or a 0.0% at the Required Speed input. The minimum on time function prevents short cycling of the outputs (speeds) when the Required Speed input is fluctuating near a speed change threshold.

Stop Sequence and Purge Time

A Fan Request of OFF or a Required Speed of 0.0% is considered a stop sequence, which causes the outputs to go OFF after completion of any Minimum On Time and/or assigned Purge Time (in seconds). The Purge Time function extends the fan's run time to purge residual air handling energy as well as prevent short cycling of the fan during sudden start / stop requests at the inputs.

If a stop sequence request occurs during an active Kick sequence, the Kick sequence completes first, and then the outputs remain active at the previously calculated level until completion of the Minimum On Time (if applicable). The outputs are then allowed to return to their digital OFF state (and Variable Speed output to 0.0%) upon completion of any assigned Purge Time.

Analog Fan Control (Variable Speed)

The Fan Speed object must be enabled (Enable set to NA or ON). A Fan Request of ON causes the Fan Speed object to use the Required Speed input value to determine the proper output response.

A fan run sequence begins when the Required Speed rises above 0.0%. Only the Variable Speed output is active in an analog configuration, all digital outputs remain OFF. The Variable Speed output reacts differently depending on the Kick sequence configuration.

Run Sequence (no Kick Time)

With no assigned Kick sequence (Kick Time = 0, the default), the Variable Speed output goes to the Required Speed input or the Start Point percent value whichever is greater. The Start Point input defines the minimum speed during any run sequence. All digital outputs remain OFF. See Table–5.132.

Run Sequence (with Kick Time)

If a Kick start sequence is assigned (Kick Time > 0), the Variable Speed output goes to 100% for the number of seconds defined in Kick Time. When this Kick timer expires, the Variable Speed output goes to the Required Speed input or the Start Point percent value whichever is greater. The Start Point input defines the minimum speed during any run sequence. All digital outputs remain OFF, as shown in Table–5.132.

Speed Change and Start Point

After a fan run sequence begins (and any Kick sequence completes), the Variable Speed output is calculated based on the Required Speed and Start Point input values. As the Required Speed input varies from .01 to 100 the Variable Speed output ranges linearly from the Start Point input value to 100. When the Required Speed input is zero the Variable Speed output is zero. Minimum On Time does not apply to a Fan Speed object configured for analog operation.

Stop Sequence and Purge Time

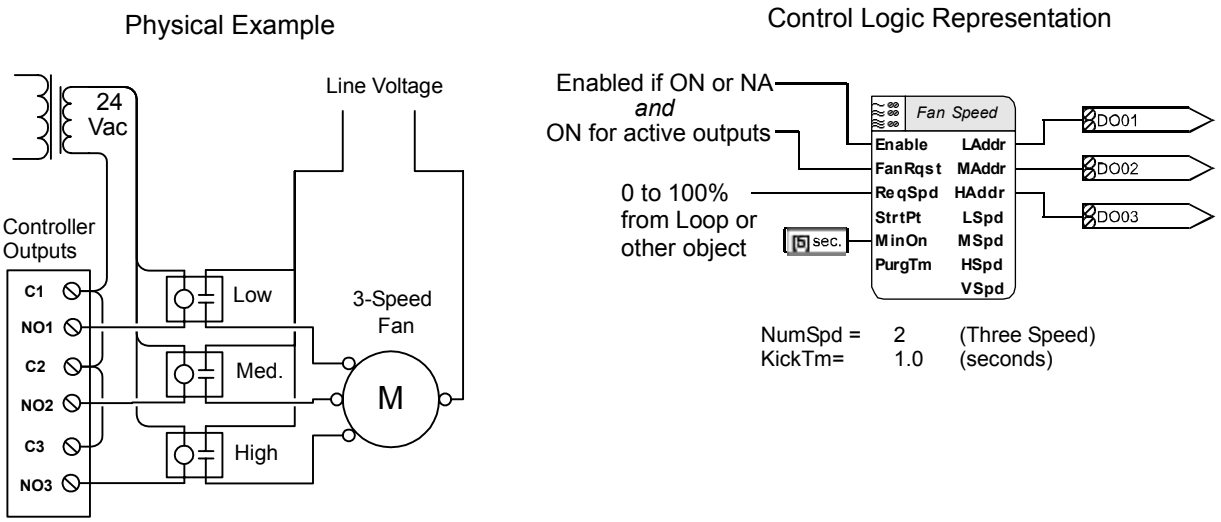
A Fan Request of OFF or a Required Speed of 0.0% is considered a stop sequence, which causes the Variable Speed output to go to the Start Point percent for the assigned Purge Time (in seconds) before ending at 0.0%. The Purge Time function extends the fan's run time to purge residual air handling energy as well as prevent short cycling of the fan during sudden start / stop requests at the inputs.

Examples

Two Fan Speed object examples follow. The first Fan Speed object controls a Three Speed fan (digital) and the second is for a Variable Speed fan (analog).

Three-Speed Fan

The Number of Speeds configuration is set to Three Speed. The Fan Speed object directly controls three digital output (DO) points on a MicroNet controller Figure-5.63. The three-speed fan has a quarter-horsepower motor that requires a “kick” start from a complete stop for 1.0 second before it can be set to low or medium speed. In addition, each speed requires a minimum 5 second ON time before changing.

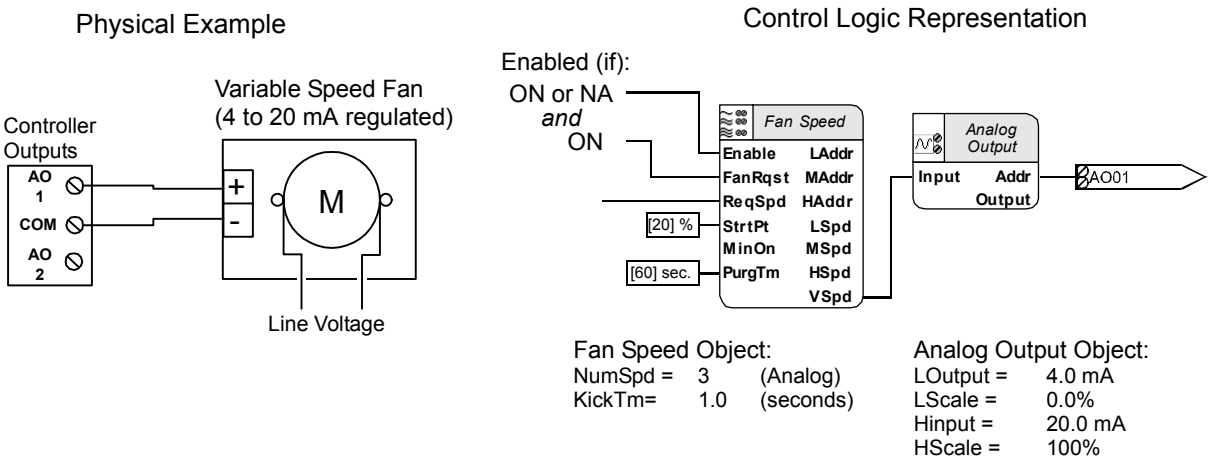


Figure–5.63 Example Fan Speed Object Used for Three-Speed Fan Control.

Variable Speed Fan

The Number of Speeds configuration is set to Analog. The Fan Speed object calculates the speed required on the Variable Speed output, which feeds an Analog Output object for hardware fan control Figure-5.64.

The variable speed fan has a third-horsepower motor that requires a “kick” start (100% output) from a complete stop for 1.0 second before it can be set to a lower speed. In addition, the minimum speed for the fan is a 20% level, represented at the Start Point input.

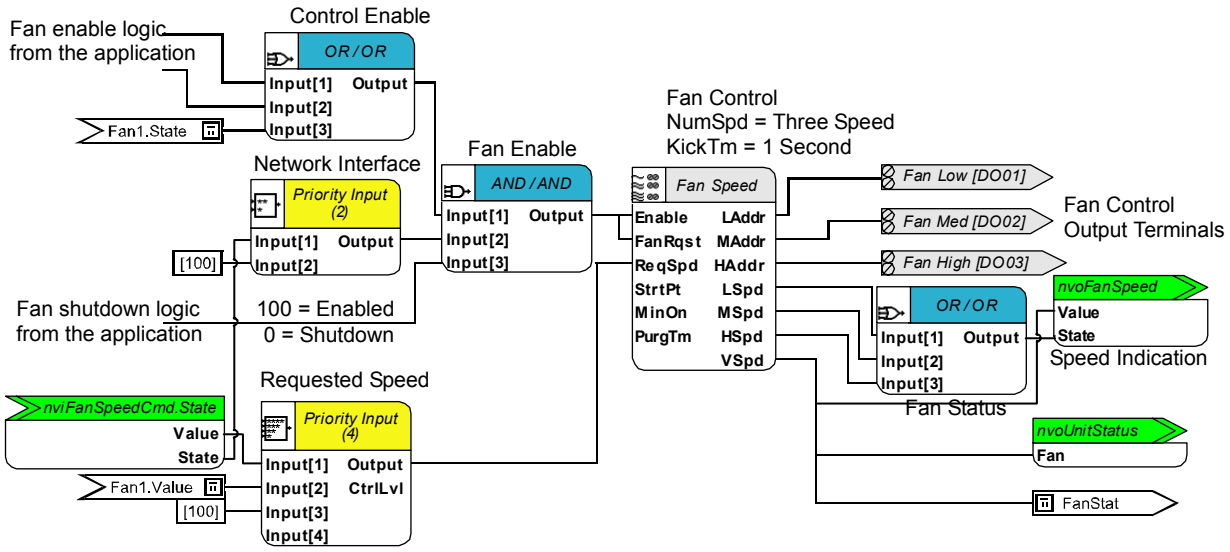


Figure–5.64 Example Fan Speed Object for Variable Speed Fan.

Any Required Speed input greater than zero results in a Variable Speed output that is greater than the Start Point value.

LON / MN-Sx Sensor Interface

The example in Figure–5.65 shows part of an application used for multi-speed fan control. Additional objects and resource tags provide fan speed interface control over the LON and with an MN-S4xx or -S5xx sensor.



Figure–5.65 Example Fan Speed Object with LON and MN-S4xx/S5xx Sensor Access.

Filter

Object Usage: The Filter object is a special-purpose math object used to dampen the rate of change for an analog value by providing an exponential low-pass filter. The filter action limits the response of the Output in relationship to a step change at the Input. The Filter object is typically used to stabilize a noisy or rapidly changing input signal, and uses the same Filter algorithm included as a configuration option for an Analog Input object.

Device Support: (all firmware revisions)

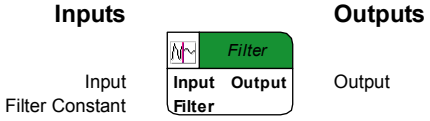
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 8 bytes
RAM: 10 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Properties

Table-5.133 Filter Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.134 Filter Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Analog - The signal to which the filter is applied.	-163.83 to 16383	A not active (NA) at the input causes the output to be set to NA
Filter	Filter Constant	Class: Analog - Defines the filter constant or filter factor applied to the input. • Maximum Filter is at 0.01. • Minimum Filter is at 0.99.	0.00 to 1.00	An not active (NA), 0.00, or 1.00 bypasses the filtering action. Values <0.00 or >1.00 are treated as 0.00 or 1.00, respectively.

Table–5.135 Filter Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output is the filtered value of the Input. If not active (NA) is present at the Input, the output is set to NA.	-163.83 to 16383

Applying the Object

A filter object is typically used to dampen the effects of a “jumping” analog value associated with a noisy or rapidly changing input signal. The filter action applied is identical to that built into the Filter configuration property of an Analog Input object. However, the filter constant used by the Filter Object is provided as an input (Filter Constant) rather than a configuration property. This provides flexibility in an online application checkout (diagnostics), as this value can be updated dynamically without an entire application recompile and download to an I/A Series MicroNet controller.

Typically, the Filter Constant value is received from a Constant value (tag) connected to the object, but may also be a value generated by other control logic for a special purpose. The range of the Filter Constant input is between 0.00 and 1.00. Input values over 1.00 are evaluated as 1.00 (no filter) and negative values are evaluated as 0.0 (also no filter). Filter Constant action is shown in Table–5.136 below.

Table–5.136 Filter Constant and Filter Action.

Filter Constant	Filter Action
0.00	No Filter
0.01 (Maximum Filter) through 0.99 (Minimum Filter)	Active Filter Area
1.00 (Default)	No Filter
Not Active (NA)	No Filter

How the Filter Algorithm Works

Between any two successive changes at the input, the filter algorithm provides the following function:

Output = Previous Output + [Filter Constant (Present Input - Previous Output)]

For example, an Analog Input object for an airflow sensor is observed to have an unstable output near the sensor's low input range, with the object output constantly jumping between 50 and 75 when airflow is holding near 60 CFM. By using a Filter Constant of 0.30, this jumping is minimized.

At first Input jump from 50 to 75:

Output = 50.00 + [0.30 (75.0 - 50.00)] or **57.50** (vs. 75.0 with No Filter)

Input jumps back to 50:

Output = 57.50 + [0.30 (50.0 - 57.50)] or **55.25** (vs. 50.0 with No Filter)

Input jumps up to 75:

Output = 55.25 + [0.30 (75.0 - 55.25)] or **61.18** (vs. 75.0 with No Filter)

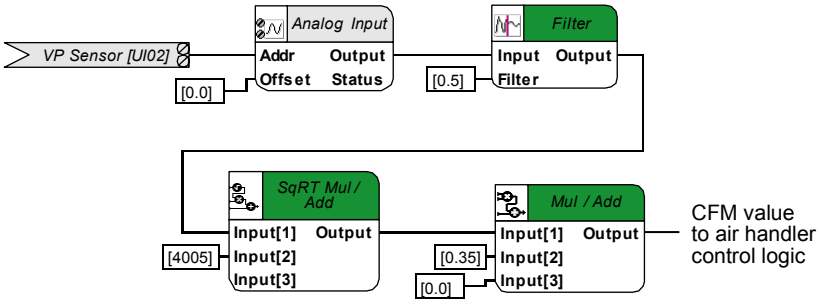
Input jumps back to 50:

Output = 61.18 + [0.30 (50.0 - 61.18)] or **57.83** (vs. 50.0 with No Filter)

And so on.....

Example Application

In the example in Figure–5.66, the Filter object is used to dampen the output value from an Analog Input object for a velocity pressure sensor. The Filter object Output feeds an input to a SqRt Mul / Add object, which calculates the current CFM airflow equivalent for the sensor.



Figure–5.66 Example Filter Object for a Velocity Pressure Sensor.

Floating Actuator

Object Usage: The Floating Actuator is a point-type object for control of a floating type actuator using two digital outputs (DOs). A single object Input receives a desired position (0 to 100%), which controls the hardware outputs. Four configurable modes allow optional use of feedback and two different types of output sequences, characterized by the state of the Drive Open and Drive Close outputs at a “hold” or setpoint control condition:

- Both outputs OFF (true floating)
- Both outputs ON (spring return)

A floating actuator without feedback must have an identical drive time for both a full open stroke and a full close stroke. This Drive Time can be from 10 seconds to a maximum of 1,000 seconds. A floating actuator with feedback must provide a scaled 0 to 100% position value (via an Analog Input object) to the Feedback input, which is used for positive positioning control for the object outputs. The Floating Actuator object includes inputs for deadband allowance and direct Force Open and Force Close overrides to the object outputs.

Device Support: (all firmware revisions)

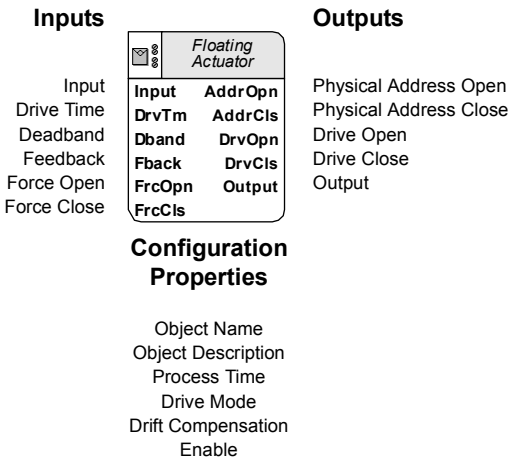
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 24 bytes
RAM: 40 bytes (standard controllers)
16 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Drive Mode
Drift Compensation Enable

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.137 Floating Actuator Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.

Table–5.137 Floating Actuator Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
DrvMode	Drive Mode	Class: Analog -Defines the mode and operating sequence of the floating actuator. • 0 = Sequence A, with *no feedback*. Both Drive Open and Drive Close outputs OFF at actuator hold control. • 1 = Sequence A, with feedback. Both Drive Open and Drive Close outputs OFF at actuator hold control. • 2 = Sequence B (spring return type) with *no feedback*. Both Drive Open and Drive Close outputs ON at actuator hold control. • 3 = Sequence B (spring return type) with feedback. Both Drive Open and Drive Close outputs ON at actuator hold control.	0	0 - Seq. A 1 - Seq. A w/ feedback 2 - Seq. B 3 - Seq. B w/ feedback	See the Sequence A and Sequence B tables Table–5.140 for complete output variations.
DrftEnb	Drift CompensationEnable	Class: Analog - Applies only for Drive Modes without feedback. An On (1) causes the drift compensation function to be enabled. An Off (0 - the default) disables the drift compensation function.	0	0 - Off 1 - On	See the “Without Feedback“ section for details on Drift Compensation.

Table–5.138 Floating Actuator Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input	Input	Class: Analog - The requested actuator position used by the object algorithm to control the physical and logical object outputs. A not active (NA) or negative value causes the outputs to drive the actuator to the closed or 0.0% position.	0 to 100%	
DrvTm	Drive Time	Class: Analog - The full stroke travel time of the actuator in seconds. A value less than 10 seconds or not active (NA) defaults the Drive Time to 0 seconds.	10 to 1,000 seconds	If the Drive Time is less than 10 seconds, the outputs go the Hold State and the output is set to 0.0%.
Dband	Deadband	Class: Analog - Defines the deadband area where the active output drive is not permitted. When the difference between the requested Input value and the calculated or actual feedback exceeds this deadband region, the necessary Drive Open or Drive Close output is activated to nullify the difference.	0.0 to 50.0%	An unconnected input or not active (NA) acts as 0.0% (no deadband). Typically, a deadband is recommended for any type of actuator, to avoid premature wear from “hunting”.

Table–5.138 Floating Actuator Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selections	Notes
Fback	Feedback	Class: Analog - Represents the actual 0.0 to 100% feedback position provided by the actuator, used only in Drive Mode selections that include feedback. A not active (NA) indicates loss of feedback and results in a full close output (Drive Close ON) for the specified Drive Time, and a NA at the object Output.	0.0 to 100%	The signal supplying the feedback to the Floating Actuator object must be scaled 0 to 100% for proper actuator operation.
FrcOpn	Force Open	Class: Digital - ON forces the Floating Actuator to a full open condition regardless of the requested position at the Input. The actuator is driven open for the full Drive Time. An OFF or not active (NA) disables the Force Open request.	—	If the Force Open and Force Close inputs are ON at the same time, the Force Close function is activated.
FrcCls	Force Close	Class: Digital - ON forces the Floating Actuator to a full closed condition regardless of the requested position at the Input. The actuator is driven closed for the full Drive Time. An OFF or not active (NA) disables the Force Close request.	—	If the Force Open and Force Close inputs are ON at the same time, the Force Close function is activated.

Table–5.139 Floating Actuator Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
AddrOpn	Physical Address Open	Class: Analog - The physical hardware address (digital output terminal point on the controller) assigned to the Drive Open function. See the Note at the bottom of this table for this property's behavior when the Drive Time is 0 seconds.	Dependent on the controller platform selected.
AddrCls	Physical Address Close	Class: Analog - The physical hardware address (digital output terminal point on the controller) assigned to the Drive Close function. See the Note at the bottom of this table for this property's behavior when the Drive Time is 0 seconds.	Dependent on the controller platform selected.
DrvOpn	Drive Open	Class: Digital - If ON, indicates the active driving of the floating actuator towards the open or 100.0% position. See the Note at the bottom of this table for this property's behavior when the Drive Time is 0 seconds.	OFF (0.0) ON (100.0)
DrvCls	Drive Close	Class: Digital - If ON, indicates the active driving of the floating actuator towards the closed or 0.0% position. See the Note at the bottom of this table for this property's behavior when the Drive Time is 0 seconds.	OFF (0.0) ON (100.0)

Table–5.139 Floating Actuator Object Output Properties. (Continued)

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The current 0.0 to 100.0% position of the floating actuator, derived as follows: • If a Drive Mode configuration without feedback, a calculated position based on the Drive Time and the object algorithm. • If a Drive Mode configuration with feedback, the actual feedback received on the Feedback input to the object. If the Drive Mode configuration uses feedback and the Feedback input is not active (NA), this Output is also set to NA. See the Note at the bottom of this table for this property's behavior when the Drive Time is 0 seconds.	0.0 to 100.0%
Note: A Drive Time value of less than 10 seconds defaults the Drive Time to 0 seconds. Anytime the drive time is 0 seconds, the AddrOpn, AddrCls, DrvOpn, DrvCls, and Output properties are set to their HOLD mode values Table–5.140. Examples: *Sequence A* AddrOpn = Off AddrCls = Off DrvOpn = Off DrvCls = Off Output = 0.0% *Sequence B* AddrOpn = On AddrCls = On DrvOpn = On DrvCls = On Output = 0.0%			

Note: Direct connection of the physical hardware outputs to the Floating Actuator Priority object is recommended for improved output drive resolution, as the Drive Open and Drive Close outputs do not directly reflect the actual physical hardware output when the calculated output times are less than the actual object execution time.

Applying the Object

The Floating Actuator object is a point-type object that directly controls a floating type actuator. The Address Open and Address Closed outputs are hardware outputs, each used for a digital output (DO) on a controller, or a triac output (TO) on an MNL-V3Rxx controller. The Drive Open and Drive Close outputs correspond to the present state of each hardware output, while the Output value represents the 0 to 100% position of the actuator.

The requested 0 to 100% position is received on the Input and produces an output sequence determined by the Drive Mode configuration. Four different Drive Modes result from a mix of feedback options (without or with) and two output sequences (A or B) as shown below Figure-5.140.

Table–5.140 Drive Modes with Physical Output Sequences.

Drive Mode	Description	Sequence	Action	AddrOpn / Drive Open	AddrCls / Drive Close
0	Sequence A	A	Hold	OFF	OFF
			Drive Open	ON	OFF
1	Sequence A with Feedback		Drive Close	OFF	ON
			Not Allowed	ON	ON
2	Sequence B	B	Spring Return	OFF	OFF
			Drive Open	ON	OFF
3	Sequence B with Feedback		Drive Close	OFF	ON
			Hold	ON	ON

Sequence A is typically used on actuators that do not require a powered hold position. Sequence B is typically used on spring return actuator that do require a powered hold position.

Without Feedback (Modes 0 and 2)

The Floating Actuator object without feedback positions the actuator based upon the requested position at the Input, the actuator's full stroke Drive Time, and the assigned Deadband. The Drive Time value must accurately define the number of seconds required by the actuator to drive a full stroke. The object algorithm uses a time-based positioning algorithm to position or drive open and drive close the actuator.

Note: A Drive Time value of less than 10 seconds defaults the Drive Time to 0 seconds. *Whenever the drive time is 0 seconds*, the AddrOpn, AddrCls, DrvOpn, DrvCls, and Output properties are set to their HOLD mode values. In Mode 0 (Sequence A), this sets the AddrOpn, AddrCls, DrvOpn, and DrvCls properties to Off, and sets the Output property to its currently calculated position. In Mode 2 (Sequence B), this sets the AddrOpn, AddrCls, DrvOpn, and DrvCls properties to On, and sets the Output property to 0.0%.

The position calculation occurs with the knowledge that the actuator travels at the same constant rate in both the drive open and drive closed directions. This function will not operate properly if the actuator is unable to hold a constant position over time. The hardware outputs are driven according to the selected Drive Mode sequence A or B as shown in Table–5.140, and the calculated position is indicated at the object's Output.

Deadband: A Deadband input to the Floating Actuator object is provided to minimize the number of output changes during minor fluctuations of the Input value. This hysteresis can keep an actuator from "hunting" during typical control sequences. Deadband is an analog value expressed in the same percent used by the Input and Output, and is evaluated by the object as follows:

While	Then
Input is within Calculated Position (factoring in Deadband)	No Drive Action
Input > (Calculated Position + 1/2 Deadband)	Drive Open until Calculated Position = Input
Input < (Calculated Position - 1/2 Deadband)	Drive Close until Calculated Position = Input

Use of deadband is strongly recommended to prevent undue mechanical wear to an actuator. A typical value for Deadband is 5.0%; the maximum allowable value is 50.0%.

Reset Synchronization Cycle: After a controller reset, the actuator is driven full closed for at least the total travel (Drive Time) to synchronize and establish the actuator closed position. With synchronization complete, the actuator is positioned based upon the value requested at the Input. A requested actuator position or a Force Open / Force Close request has no effect upon the synchronization process until the reset synchronization cycle is complete.

Automatic Calibration: An internal calibration feature automatically synchronizes the actuator position whenever the actuator is commanded to the full closed (0.0%) or full open (100.0%) position by continuing to drive the actuator in the requested direction for a time period of at least one additional complete full stroke drive time. If during this “end of stroke” drive period the Input value changes to a value within the control range, the output drive to locate the true end of stroke operation is terminated.

Drift Compensation Enable: Setting the Drift Compensation Enable configuration property to On (1) compensates for floating actuator mechanical drift whenever the actuator remains at the full open or full closed position for extended periods of time. Drift compensation is done by periodically driving the actuator in the commanded direction for a time period derived from the Drive Time assigned. Drift compensation does not occur whenever the actuator is within the actual control range between full open and full closed.

For example, an assigned Drive Time of 60 seconds causes the drift compensation algorithm to operate every 600 seconds (10 minutes) and drive the actuator in the appropriate direction for 6 seconds.

Drift compensation (Period) = 10 X Drive Time

Drift compensation (Drive Time) = Drive Time ÷ 10

Note: On actuators that contain an onboard hardware minimum-position setting, set the onboard actuator minimum-position to 0%, and allow the application to perform a minimum position function.

Force Open / Force Close Overrides: The Force Open / Force Close inputs can be used to override the floating actuator to a full open or full closed condition regardless of the requested position at the Input.

- An ON at the Force Open input results in a full open output sequence (AddrOpn / Drive Open are ON for the Drive Time period.)
  An OFF or not active (NA) disables the Force Open request.
- An ON at the Force Close input results in a full close output sequence (AddrCls / Drive Close are ON for the Drive Time period.)
  An OFF or not active (NA) disables the Force Close request.

Note: A simultaneous ON at both the Force Close and Force Open inputs results in the Force Close function.

NA Input: If the Input goes to a not active (NA) condition, the object drives the outputs to the closed or 0.0% position. Note, however, that the Force Close and Force Open inputs remain functional.

With Feedback (Modes 1 and 3)

The Floating Actuator object compares the Feedback value to the requested position at the Input and calculates a required drive time and direction used to activate the necessary hardware output(s), until the Feedback matches the Input value requested.

Note: A Floating Actuator object with feedback requires an active 0 to 100% feedback value present at the Feedback input. In Modes 1 and 3, the feedback signal is typically provided by a separate Analog Input object scaled 0 to 100% for proper actuator operation.

The hardware outputs are driven according to the selected Drive Mode sequence A or B Figure-5.140, and the Feedback position value is reflected at the object's Output. The Feedback signal provides a positive (rather than a calculated) actuator position. As a result, automatic compensation, drift compensation, and controller reset routines are not used when controlling with feedback. The full stroke time for the actuator with feedback should still be assigned to the drive time input. Drive time will be used as a safety or confirmation of actuator drive during actuator positioning.

Note: A Drive Time value of less than 10 seconds defaults the Drive Time to 0 seconds. *Whenever the drive time is 0 seconds*, the AddrOpn, AddrCls, DrvOpn, DrvCls, and Output properties are set to their HOLD mode values. In Mode 1 (Sequence A with feedback), this sets the AddrOpn, AddrCls, DrvOpn, and DrvCls properties to Off, and sets the Output property to its currently calculated position. In Mode 3 (Sequence B with feedback), this sets the AddrOpn, AddrCls, DrvOpn, and DrvCls properties to On, and sets the Output property to 0.0%.

Deadband: A Deadband input to the Floating Actuator object is provided to minimize the number of output changes during minor fluctuations of the Input value. This hysteresis can keep an actuator from "hunting" during typical control sequences. Deadband is an analog value expressed in the same percent used by the Input and Output, and is evaluated by the object as follows:

While	Then
Input is within Feedback (factoring in Deadband)	No Drive Action
Input > (Feedback + 1/2 Deadband)	Drive Open until Feedback = Input
Input < (Feedback - 1/2 Deadband)	Drive Close until Feedback = Input

The Floating Actuator object algorithm internally limits the deadband to a percentage which prevents output changes smaller than one second. For example, a Drive Time of 135 seconds will internally limit the deadband to: [(1 second ÷ 135 seconds) x 2], or 1.48%. The object algorithm uses the internally calculated deadband if the user-assigned Deadband is less than the calculated limit.

Drive Timeout: When commanded to a new Input position, the appropriate hardware output is activated until either the Feedback reaches the correct position or a timeout exceeding the Drive Time expires. An exceeded timeout indicates the actuator is unable to achieve the requested position. The object then sets both hardware outputs to OFF until the object attempts to drive the actuator in the opposite direction. The following conditions cause the object to reverse the actuator and clear the drive timeout.

- If the Drive Open output was ON when the Drive Time expired, the object must attempt to drive the actuator to the closed position to clear the drive timeout. This occurs when the Input decreases to a value approximately equal to the Feedback input minus one half of the Deadband value. This also occurs when the Feedback increases to a value approximately equal to the Input plus one half of the Deadband value.
- If the Drive Close output was ON when the Drive Time expired, the object must attempt to drive the actuator to the open position to clear the drive timeout. This occurs when the Input increases to a value approximately equal to the Feedback input plus one half of the Deadband value. This also occurs when the Feedback input decreases to a value approximately equal to the Input minus one half of the Deadband value.

NA Input: If the Input goes to a not active (NA) condition, the object drives the outputs to the closed or 0.0% position. Note, however, that the Force Close and Force Open inputs remain functional.

Feedback Fault: A not active (NA) at the Feedback input indicates the loss of a valid feedback signal causing the object to drive towards the full closed position (ClsAddr / Drive Close to ON) for the specified Drive Time. The object Output value also remains at NA while the Feedback value is at NA.

Force Open / Force Close Overrides: The Force Open / Force Close inputs can be used to override the floating actuator to a full open or full closed condition regardless of the requested position at the Input.

- An ON at the Force Open input results in a full open output sequence (AddrOpn / Drive Open are ON for the Drive Time period.)
  An OFF or not active (NA) disables the Force Open request.
- An ON at the Force Close input results in a full close output sequence (AddrCls / Drive Close are ON for the Drive Time period.)
  An OFF or not active (NA) disables the Force Close request.

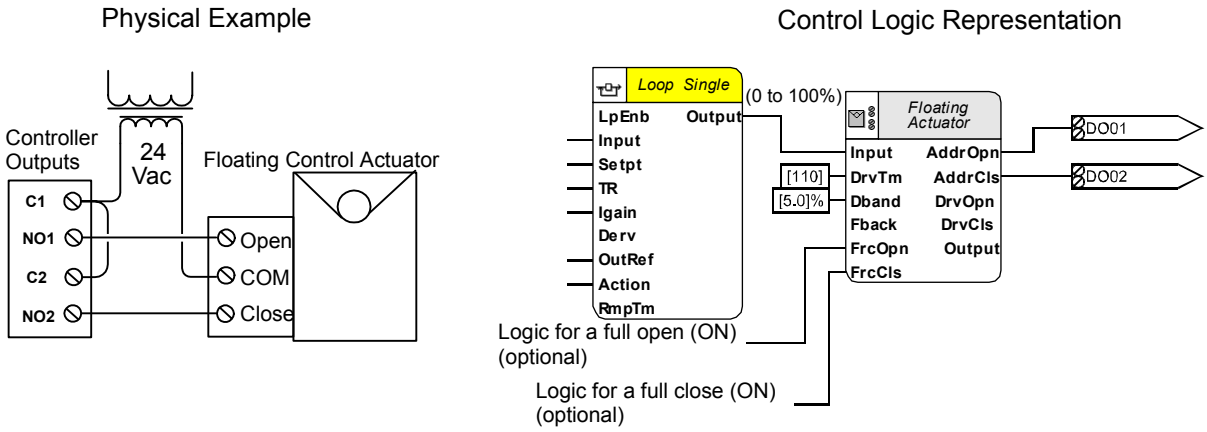
Note: A simultaneous ON at both the Force Close and Force Open inputs results in the Force Close function.

Examples

Two Floating Actuator object examples follow. The first example is for controlling an actuator without feedback Figure-5.67. The second example is for an actuator that has feedback Figure-5.68.

Without Feedback

A floating actuator with a drive time of 110 seconds is used to modulate an outdoor air damper. The actuator has no provision for feedback.

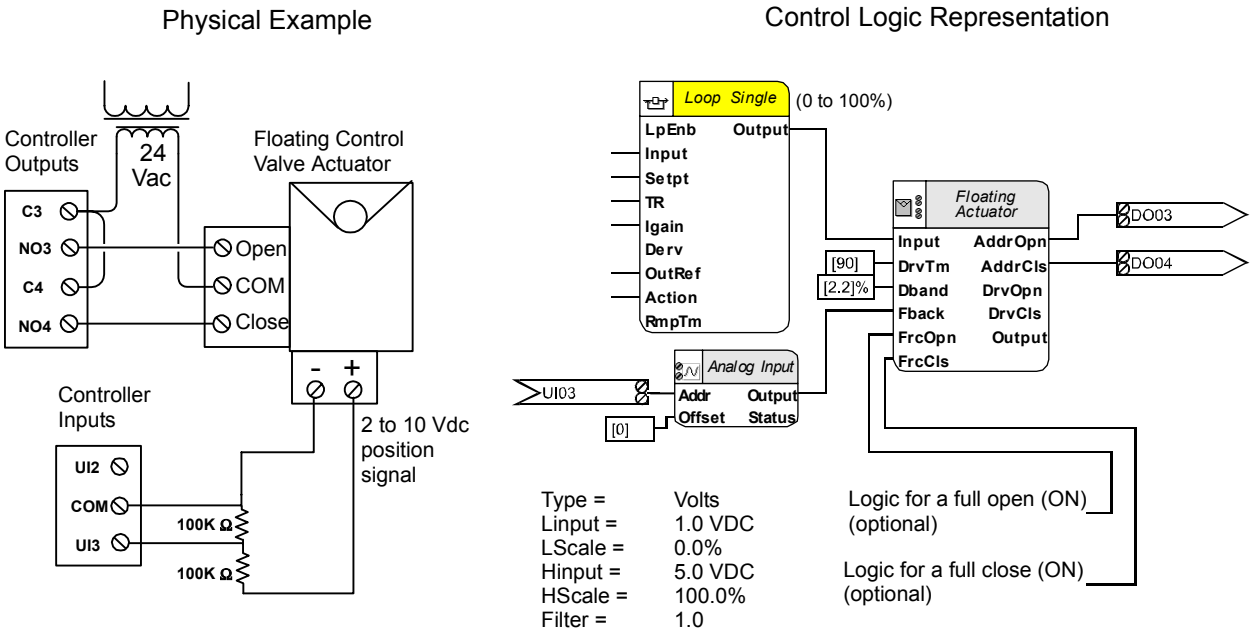


Figure–5.67 Floating Actuator Object Example for an Actuator Without Feedback.

With Feedback

A floating actuator with a voltage feedback option is used to modulate a hot water supply valve. The actuator has a full-stroke drive time of 90 seconds.

An Analog Input object voltage-divider combination is used to accept the 2 to 10 Vdc hardware feedback signal provided by the actuator. The voltage-divider is necessary to reduce the feedback signal to within the hardware input range of the controller input (0 to 5 Vdc), in this case, 1 to 5 Vdc. The Analog Input object is scaled to convert the 1 to 5 Vdc signal to an analog value between 0 and 100%.



Figure–5.68 Floating Actuator Object Example for an Actuator With Feedback.

The scaled output of the Analog Input object connects to the Feedback input of the Floating Actuator object Figure-5.68, to provide current valve position. This Feedback value is reflected at the Output of the Floating Actuator object.

Note: A Floating Actuator object with feedback requires an active 0 to 100% feedback value present at the Feedback input. In Modes 1 and 3, the feedback signal is typically provided by a separate Analog Input object scaled 0 to 100% for proper actuator operation.

Floating Actuator Priority

Object Usage: The Floating Actuator Priority object is a point-type object for control of a floating type actuator using two digital outputs (DOs). The object functions exactly like the Floating Actuator object, but features four prioritized inputs that can each receive a desired position (0 to 100%). The highest valid input controls the hardware outputs. Four configurable modes allow optional use of feedback and two different types of output sequences, characterized by the state of the Drive Open and Drive Close outputs at a “hold” or setpoint control condition:

- Both outputs OFF (true floating)
- Both outputs ON (spring return)

A floating actuator without feedback must have an identical drive time for both a full open stroke and a full close stroke. This Drive Time can be from 10 seconds to a maximum of 1,000 seconds. A floating actuator with feedback must provide a scaled 0 to 100% position value (via an Analog Input object) to the Feedback input, which is used for positive positioning control for the object outputs. The Floating Actuator Priority object includes inputs for deadband allowance and direct Force Open and Force Close overrides to the object outputs.

Device Support: (all firmware revisions)

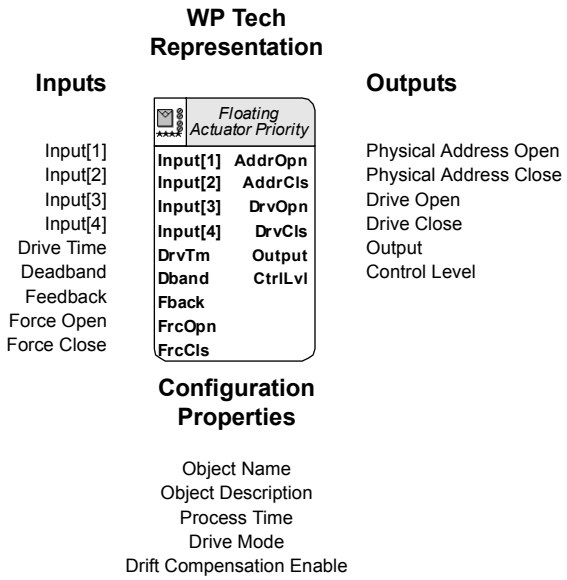
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 30 bytes
RAM: 48 bytes (standard controllers)
18 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Drive Mode
Drift Compensation Enable

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.141 Floating Actuator Priority Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
DrvMode	Drive Mode	Class: Analog - Defines the mode and operating sequence of the floating actuator. • 0 = Sequence A, with *no feedback*. Both Drive Open and Drive Close outputs OFF at actuator hold control. • 1 = Sequence A, with feedback. Both Drive Open and Drive Close outputs OFF at actuator hold control. • 2 = Sequence B (spring return type) with *no feedback*. Both Drive Open and Drive Close outputs ON at actuator hold control. • 3 = Sequence B (spring return type) with feedback. Both Drive Open and Drive Close outputs ON at actuator hold control.	0	0 - Seq. A 1 - Seq. A w/ feedback 2 - Seq. B 3 - Seq. B w/ feedback	See the Sequence A and Sequence B tables for complete output variations.
DrftEnb	Drift CompensationEnable	Class: Analog - Applies only for Drive Modes without feedback. An On (1) causes the drift compensation function to be enabled. An Off (0 - the default) disables the drift compensation function.	0	0 - Off 1 - On	See the “Without Feedback“ section for details on Drift Compensation.

Table–5.142 Floating Actuator Priority Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input[1]	Input[1]	Class: Analog - The requested actuator position with the highest priority. This input is monitored first to control the physical and logical object outputs.	0.0 to 100.0%	A not active (NA) at this input causes the second input to be evaluated for a valid value.
Input[2]	Input[2]	Class: Analog - The requested actuator position with the second highest priority. This input is monitored if Input[1] has a NA, and is used to control the physical and logical object outputs.	0.0 to 100.0%	A not active (NA) at this input causes the third input to be evaluated for a valid value.

Table–5.142 Floating Actuator Priority Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input[3]	Input[3]	Class: Analog - The requested actuator position with the third highest priority.This input is monitored if Inputs[1] and [2] are both NA, and is used to control the physical and logical object outputs.	0.0 to 100.0%	A not active (NA) at this input causes the fourth and last input to be evaluated for a valid value.
Input[4]	Input[4]	Class: Analog - The requested actuator position with the lowest priority.This input is monitored if all other Inputs have a not active (NA), and is used to control the physical and logical object outputs.	0.0 to 100.0%	If all inputs including Input[4] have a not active (NA), the hardware and Drive Close outputs drive the actuator to the closed or 0.0% position.
DrvTm	Drive Time	Class: Analog - The full stroke travel time of the actuator in seconds. A value less than 10 seconds or not active (NA) defaults the Drive Time to 0 seconds.	10 to 1,000 seconds	If the Drive Time is less than 10 seconds, the outputs go to the Hold State and the output is set to 0.0%
Dband	Deadband	Class: Analog - Defines the deadband area where the active output drive is not permitted. When the difference between the requested Input value and the calculated or actual feedback exceeds this deadband region, the necessary Drive Open or Drive Close output is activated to nullify the difference.	0.0 to 50.0%	An unconnected input or not active (NA) acts as 0.0% (no deadband). Typically, deadband is recommended for any floating type of actuator, to prevent premature wear from “hunting”.
Fback	Feedback	Class: Analog - Represents the actual 0.0 to 100% feedback position provided by the actuator, used only in Drive Mode selections that include feedback. A not active (NA) indicates loss of feedback and results in a full close output (Drive Close ON) for the specified Drive Time, and a NA at the object Output.	0.0 to 100%	The signal supplying the feedback to the Floating Actuator Priority object must be scaled 0 to 100% for proper actuator operation.
FrcOpn	Force Open	Class: Digital - ON forces the Floating Actuator priority to a full open condition regardless of the requested position at the Input. The actuator is driven open for the full Drive Time. An OFF or not active (NA) disables the Force Open request.	—	If the Force Open and Force Close inputs are ON at the same time, the Force Close function is activated.
FrcCls	Force Close	Class: Digital - ON forces the Floating Actuator priority to a full closed condition regardless of the requested position at the Input. The actuator is driven closed for the full Drive Time. An OFF or not active (NA) disables the Force Close request.	—	If the Force Open and Force Close inputs are ON at the same time, the Force Close function is activated.

Table–5.143 Floating Actuator Priority Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
AddrOpn	Physical Address Open	Class: Analog - The physical hardware address (digital output terminal point on the controller) assigned to the Drive Open function. See the Note at the bottom of this table for this property’s behavior when the Drive Time is 0 seconds.	Dependent on the controller platform selected.

Table–5.143 Floating Actuator Priority Object Output Properties. (Continued)

Abbrev.	Name	Class / Description	Valid Values
AddrCls	Physical Address Close	Class: Analog - The physical hardware address (digital output terminal point on the controller) assigned to the Drive Close function. See the Note at the bottom of this table for this property's behavior when the Drive Time is 0 seconds.	Dependent on the controller platform selected.
DrvOpn	Drive Open	Class: Digital - If ON, indicates the active driving of the floating actuator towards the open or 100.0% position. See the Note at the bottom of this table for this property's behavior when the Drive Time is 0 seconds.	OFF (0.0) ON (100.0)
DrvCls	Drive Close	Class: Digital - If ON, indicates the active driving of the floating actuator towards the closed or 0.0% position. See the Note at the bottom of this table for this property's behavior when the Drive Time is 0 seconds.	OFF (0.0) ON (100.0)
Output	Output	Class: Analog - The current 0.0 to 100.0% position of the floating actuator, derived as follows: • If a Drive Mode configuration without feedback is selected, this is a calculated position based on the Drive Time and the object algorithm. • If a Drive Mode configuration with feedback is selected, this is the actual feedback received on the Feedback input to the object. If the Drive Mode configuration uses feedback and the Feedback input is not active (NA), this Output is also set to NA. See the Note at the bottom of this table for this property's behavior when the Drive Time is 0 seconds.	0.0 to 100.0%
CtrlLvl	Control Level	Class: Analog - Indicates the currently active input by providing the priority number of the related input, that is 1, 2, 3, or 4. If all four inputs have a not active (NA), this output also goes to NA.	1, 2, 3, or 4

Note: A Drive Time value of less than 10 seconds defaults the Drive Time to 0 seconds. Anytime the drive time is 0 seconds, the AddrOpn, AddrCls, DrvOpn, DrvCls, and Output properties are set to their HOLD mode values Table–5.144. Examples:

Sequence A	*Sequence B*
AddrOpn = Off	AddrOpn = On
AddrCls = Off	AddrCls = On
DrvOpn = Off	DrvOpn = On
DrvCls = Off	DrvCls = On
Output = 0.0%	Output = 0.0%

Note: Direct connection of the physical hardware outputs to the Floating Actuator Priority object is recommended for improved output drive resolution, as the Drive Open and Drive Close outputs do not directly reflect the actual physical hardware output when the calculated output times are less than the actual object execution time.

Applying the Object

The Floating Actuator Priority object is a point-type object that directly controls a floating type actuator. The Address Open and Address Closed outputs are hardware outputs, each used for a digital output (DO) on a controller, or a triac output (TO) on an MNL-V3Rxx controller. The Drive Open and Drive Close outputs correspond to the present state of each hardware output, while the Output value represents the 0 to 100% position of the actuator.

The requested 0 to 100% position is received on the highest priority input (Input[1 - 4]) and produces an output sequence determined by the Drive Mode configuration. Four different Drive Modes result from a mix of feedback options (without or with) and two output sequences (A or B) as shown below Figure-5.144.

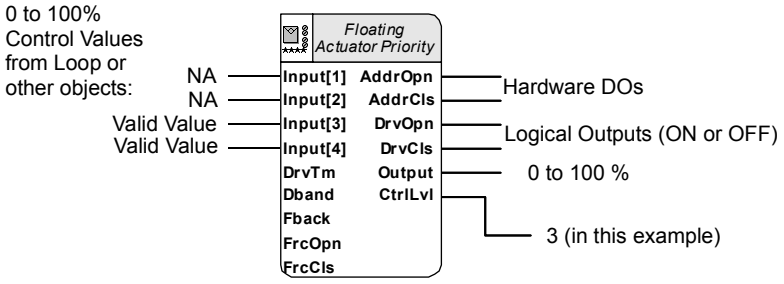
Table–5.144 Floating Actuator Priority Object Drive Modes with Physical Output Sequences.

Drive Mode	Description	Sequence	Action	AddrOpn / Drive Open	AddrCls / Drive Close
0	Sequence A	A	Hold	OFF	OFF
			Drive Open	ON	OFF
1	Sequence A with Feedback		Drive Close	OFF	ON
			Not Allowed	ON	ON
2	Sequence B	B	Spring Return	OFF	OFF
			Drive Open	ON	OFF
3	Sequence B with Feedback		Drive Close	OFF	ON
			Hold	ON	ON

Sequence A is typically used on actuators that do not require a powered hold position. Sequence B is typically used on spring return actuators that do require a powered hold position.

Priority Inputs and Values

Input[1] is the highest priority input, and is always evaluated first on each scan of the inputs. Any valid value present on Input[1] becomes the Input to the object, regardless of the state of the other inputs. A valid value is any numeric value besides a not active [NA].



Figure–5.69 Input[3] as the Current Active Input.

If Input[1] has an NA, then Input[2] is evaluated in the same manner. This priority scan continues only if Input[2] also has an NA, at which point Input[3] is evaluated, and if Input[3] also has an NA, to lastly evaluate Input[4]. If Input[4] also has an NA, the actuator is driven closed position. Control Level and Output will be set to NA.

Typically, input values are within a normal range, that is, between 0.0 and 100.0%. However, any value outside this range is evaluated as either 0.0 or 100.0. For example, a value of 165.0 is evaluated as 100.0. Likewise, a negative value such as - 56.7 would be evaluated by the object as 0.0.

Without Feedback (Modes 0 and 2)

The Floating Actuator Priority object without feedback positions the actuator based upon the requested position at the highest valid Input[1] - [4], the actuator's full stroke Drive Time, and the assigned Deadband.
The Drive Time value must accurately define the number of seconds required by the actuator to drive a full stroke. The object algorithm uses a time-based positioning algorithm to position, or drive open and drive close, the actuator.

Note: A Drive Time value of less than 10 seconds defaults the Drive Time to 0 seconds. *Whenever the drive time is 0 seconds*, the AddrOpn, AddrCls, DrvOpn, DrvCls, and Output properties are set to their HOLD mode values. In Mode 0 (Sequence A), this sets the AddrOpn, AddrCls, DrvOpn, and DrvCls properties to Off, and sets the Output property to its currently calculated position. In Mode 2 (Sequence B), this sets the AddrOpn, AddrCls, DrvOpn, and DrvCls properties to On, and sets the Output property to 0.0%

The position calculation occurs with the knowledge that the actuator travels at the same constant rate in both the drive open and drive closed directions. This function will not operate properly if the actuator is unable to hold a constant position over time. The hardware outputs are driven according to the selected Drive Mode sequence A or B Figure-5.140, and the calculated position is indicated at the object's Output.

Deadband: A Deadband input to the Floating Actuator Priority object is provided to minimize the number of output changes during minor fluctuations of the Input value. This hysteresis can keep an actuator from "hunting" during typical control sequences. Deadband is an analog value expressed in the same percent used by the Input and Output, and is evaluated by the object as follows:

While	Then
Input is within Calculated Position (factoring in Deadband)	No Drive Action
Input > (Calculated Position + 1/2 Deadband)	Drive Open until Calculated Position = Input
Input < (Calculated Position - 1/2 Deadband)	Drive Close until Calculated Position = Input

Use of deadband is strongly recommended to prevent undue mechanical wear to an actuator. A typical value for Deadband is 5.0%; the maximum allowable value is 50.0%.

Reset Synchronization Cycle: After a controller reset, the actuator is driven full closed for at least the total travel (Drive Time) to synchronize and establish the actuator closed position. With synchronization complete, the actuator is positioned based upon the value requested at the Input. A requested actuator position or a Force Open / Force Close request has no effect upon the synchronization process until the reset synchronization cycle is complete.

Automatic calibration: An internal calibration feature automatically synchronizes the actuator position whenever the actuator is commanded to the full closed (0.0%) or full open (100.0%) position by continuing to drive the actuator in the requested direction for a time period of at least one additional complete full stroke drive time. If during this “end of stroke” drive period the Input value changes to a value within the control range, the output drive to locate the true end of stroke operation is terminated.

Drift Compensation Enable: Setting the Drift Compensation Enable configuration property to On (1) compensates for floating actuator mechanical drift whenever the actuator remains at the full open or full closed position for extended periods of time. Drift compensation is done by periodically driving the actuator in the commanded direction for a time period derived from the Drive Time assigned. Drift compensation does not occur whenever the actuator is within the actual control range between full open and full closed.

For example, an assigned Drive Time of 60 seconds causes the drift compensation algorithm to operate every 600 seconds (10 minutes) and drive the actuator in the appropriate direction for 6 seconds.

Drift compensation (Period) = 10 X Drive Time

Drift compensation (Drive Time) = Drive Time ÷ 10

Note: On actuators that contain an onboard hardware minimum position-setting, set the onboard actuator minimum-position to 0%, and allow the application to perform a minimum position function.

Force Open / Force Close Overrides: The Force Open / Force Close inputs can be used to override the floating actuator to a full open or full closed condition regardless of the requested position at the Input.

- An ON at the Force Open input results in a full open output sequence (AddrOpn / Drive Open are ON for the Drive Time period plus the time required to move to the calculated 100.0% output position.)
  An OFF or not active (NA) disables the Force Open request.
- An ON at the Force Close input results in a full close output sequence (AddrCls / Drive Close are ON for the Drive Time period plus the time required to move to the calculated 0.0% output position.)
  An OFF or not active (NA) disables the Force Close request.

Note: A simultaneous ON at both the Force Close and Force Open inputs results in the Force Close function.

NA Input: If all inputs (1 to 4) are found not active (NA), the object drives the outputs to the closed or 0.0% position. Note, however, that the Force Close and Force Open inputs remain functional.

With Feedback (Modes 1 and 3)

The Floating Actuator Priority object compares the Feedback value to the requested position at the Input and calculates a required drive time and direction used to activate the necessary hardware output(s), until the Feedback matches the Input value requested.

Note: A Floating Actuator Priority object with feedback requires an active 0 to 100% feedback value present at the Feedback input. In Modes 1 and 3, the feedback signal is typically provided by a separate Analog Input object scaled 0 to 100% for proper actuator operation.

The hardware outputs are driven according to the selected Drive Mode sequence A or B Figure-5.140, and the Feedback position value is reflected at the object's Output. The Feedback signal provides a positive (rather than a calculated) actuator position. As a result, automatic compensation, drift compensation, and controller reset routines are not used when controlling with feedback. The full stroke time for the actuator with feedback should still be assigned to the drive time input. Drive time will be used as a safety or confirmation of actuator drive during actuator positioning.

Note: A Drive Time value of less than 10 seconds defaults the Drive Time to 0 seconds. *Whenever the drive time is 0 seconds*, the AddrOpn, AddrCls, DrvOpn, DrvCls, and Output properties are set to their HOLD mode values. In Mode 1 (Sequence A with feedback), this sets the AddrOpn, AddrCls, DrvOpn, and DrvCls properties to Off, and sets the Output property to its currently calculated position. In Mode 3 (Sequence B with feedback), this sets the AddrOpn, AddrCls, DrvOpn, and DrvCls properties to On, and sets the Output property to 0.0%.

Deadband: A Deadband input to the Floating Actuator Priority object is provided to minimize the number of output changes during minor fluctuations of the Input value. This hysteresis can keep an actuator from "hunting" during typical control sequences. Deadband is an analog value expressed in the same percent used by the Input and Output, and is evaluated by the object as follows:

While	Then
Input is within Feedback (factoring in Deadband)	No Drive Action
Input > (Feedback + 1/2 Deadband)	Drive Open until Feedback = Input
Input < (Feedback - 1/2 Deadband)	Drive Close until Feedback = Input

The Floating Actuator Priority object algorithm internally limits the deadband to a percentage which prevents output changes smaller than one second. For example, a Drive Time of 135 seconds will internally limit the deadband to: [(1 second ÷ 135 seconds) x 2], or 1.48%. The object algorithm uses the internally calculated deadband if the user-assigned Deadband is less than the calculated limit.

Drive Timeout: When commanded to a new Input position, the appropriate hardware output is activated until either the Feedback reaches the correct position or a timeout exceeding the Drive Time expires. An exceeded timeout indicates the actuator is unable to achieve the requested position. The object then sets both hardware outputs to OFF until the object attempts to drive the actuator in the opposite direction. The following conditions cause the object to reverse the actuator and clear the drive timeout.

- If the Drive Open output was ON when the Drive Time expired, the object must attempt to drive the actuator to the closed position to clear the drive timeout. This occurs when the highest priority level Input decreases to a value approximately equal to the Feedback input minus one half of the Deadband value. This also occurs when the Feedback input increases to a value approximately equal to the highest level priority Input plus one half of the Deadband value.
- If the Drive Close output was ON when the Drive Time expired, the object must attempt to drive the actuator to the open position to clear the drive timeout. This occurs when the highest priority level Input increases to a value approximately equal to the Feedback input plus one half of the Deadband value. This also occurs when the Feedback Input decreases to a value approximately equal to highest level priority Input minus one half of the Deadband value.

NA Input: If all inputs (1 to 4) are found not active (NA), the object drives the outputs to the closed or 0.0% position. Note, however, that the Force Close and Force Open inputs remain functional.

Feedback Fault: A not active (NA) at the Feedback input indicates the loss of a valid feedback signal causing the object to drive towards the full closed position (ClsAddr / Drive Close to ON) for the specified Drive Time. The object Output value also remains at NA while the Input value is at NA.

Force Open / Force Close Overrides: The Force Open / Force Close inputs can be used to override the floating actuator to a full open or full closed condition regardless of the requested position at the Input.

- An ON at the Force Open input results in a full open output sequence (AddrOpn / Drive Open are ON for the Drive Time period).
  An OFF or not active (NA) disables the Force Open request.
- An ON at the Force Close input results in a full close output sequence (AddrCls / Drive Close are ON for the Drive Time period).
  An OFF or not active (NA) disables the Force Close request.

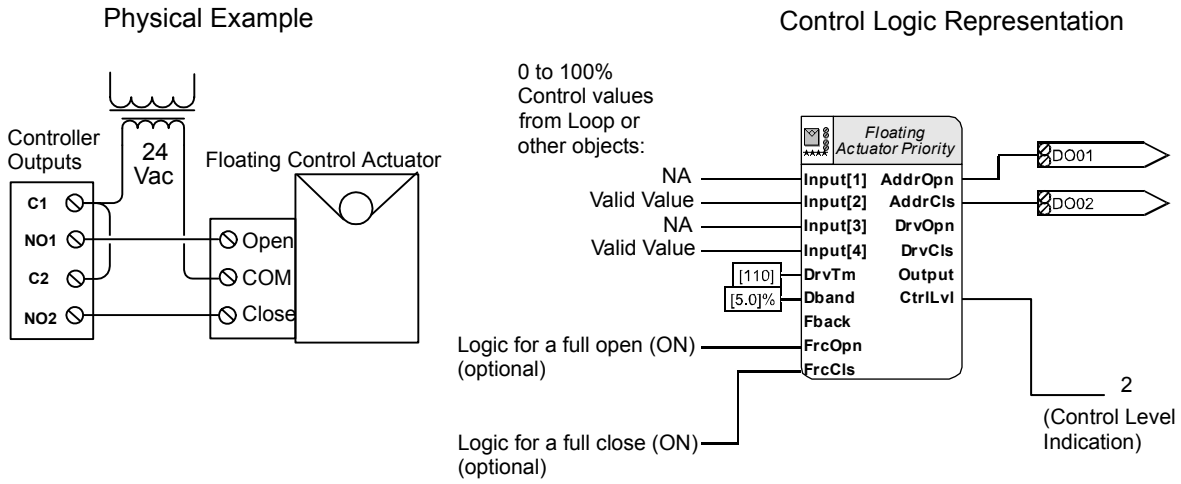
Note: A simultaneous ON at both the Force Close and Force Open inputs results in the Force Close function.

Examples

Two Floating Actuator Priority object examples follow. The first example is for controlling an actuator without feedback Figure-5.70. The second example is for an actuator with feedback Figure-5.71.

Without Feedback

A floating actuator with a drive time of 110 seconds is used to modulate an outdoor air damper. The actuator has no provision for feedback.

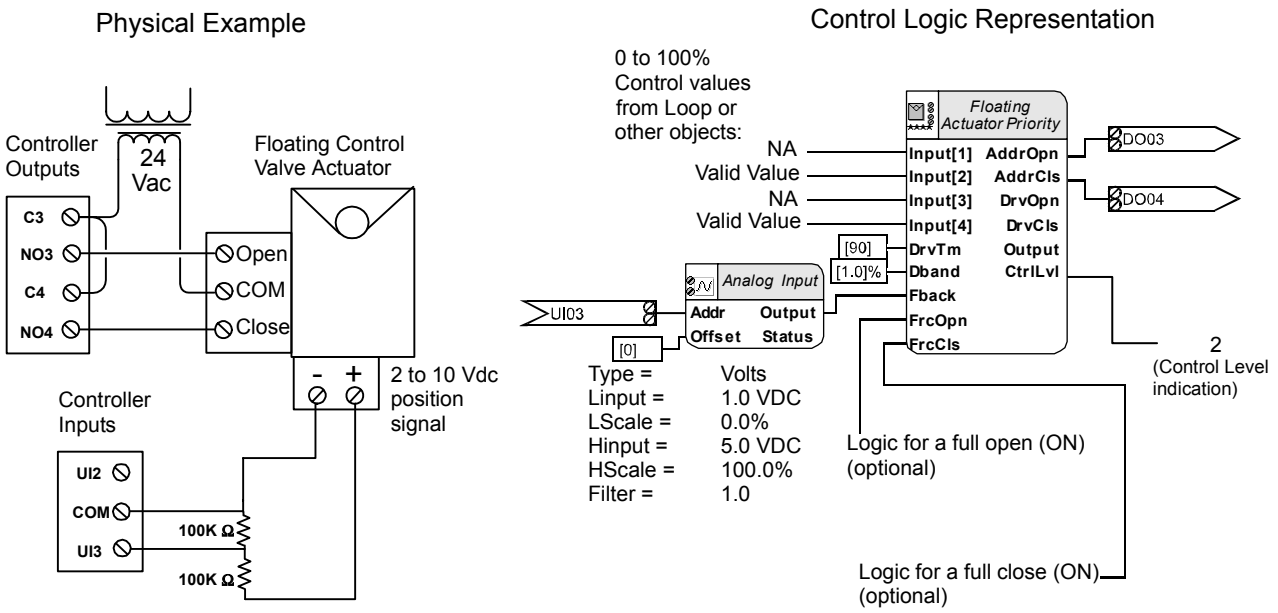


Figure–5.70 Floating Actuator Priority Object Example for an Actuator Without Feedback.

With Feedback

A floating actuator with a voltage feedback option is used to modulate a hot water supply valve. The actuator has a full-stroke drive time of 90 seconds.

An Analog Input object voltage-divider combination is used to accept the 2 to 10 Vdc hardware feedback signal provided by the actuator. The voltage-divider is necessary to reduce the feedback signal to within the hardware input range of the controller input (0 to 5 Vdc), in this case, 1 to 5 Vdc. The Analog Input object must be scaled to convert the 1 to 5 Vdc signal to an analog value between 0 and 100%.



Figure–5.71 Floating Actuator Priority Object Example for an Actuator With Feedback.

The scaled output of the Analog Input object connects to the Feedback input of the Floating Actuator Priority object Figure-5.71, to provide current valve position. This Feedback value is reflected at the Output of the Floating Actuator Priority object.

The active priority input (1, 2, 3, or 4) is identified at the Control Level output, as shown in the figure.

Note: A Floating Actuator Priority object with feedback requires an active 0 to 100% feedback value present at the Feedback input. In Modes 1 and 3, the feedback signal is typically provided by a separate Analog Input object scaled 0 to 100% for proper actuator operation.

High Select

Object Usage: The High Select object acts as a high signal selector which outputs the highest value present on the three inputs. The High Select object is typically used with analog values.

Output = Highest Input[1], [2], or [3]

The High Select object is the functional opposite of the Low Select object.

Device Support: (all firmware revisions)

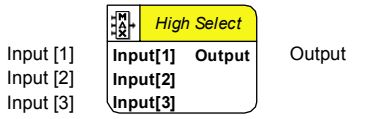
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.145 High Select Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.146 High Select Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The first input evaluated for the highest value. A not active (NA) is not evaluated.	-163.83 to 16383	If not active (NA) is present at all Inputs, the output is set to NA.
Input[2]	Input [2]	Class: Analog - The second input evaluated for the highest value. A not active (NA) is not evaluated.	-163.83 to 16383	
Input[3]	Input [3]	Class: Analog - The third input evaluated for the highest value. A not active (NA) is not evaluated.	-163.83 to 16383	

Table–5.147 High Select Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output reflects the highest input value. A value of not active (NA) indicates that all inputs are NA.	-163.83 to 16383

Applying the Object

The High Select object outputs the highest valid value present on the inputs. Not active (NA) inputs are ignored unless all inputs are NA, in which case the Output is NA. The following truth table Figure-5.148 shows all possible input to output combinations:

Table–5.148 Truth Table for High Select Object.

Input[1]	Input[2]		Output
AV1	AV2	AV3	Highest of (AV1, AV2, AV3)
AV1	AV2	NA	Highest of (AV1, AV2)
AV1	NA	AV3	Highest of (AV1, AV3)
NA	AV2	AV3	Highest of (AV2, AV3)
AV1	NA	NA	AV1
NA	AV2	NA	AV2
NA	NA	AV3	AV3
NA	NA	NA	NA

Inputs are typically analog values provided by another object's output or from an assigned constant. However, Inputs can also process numerical representations of digital values (0 for OFF or 100 for ON).

Interlock

Object Usage: The Interlock object compares a digital ON command at the Input to an independent Feedback input to determine if a valid control state exists. The feedback signal must match the commanded input ON within the assigned interlock Delay Time (in seconds) or an interlock failure is indicated at the object outputs. Typical applications include proof of fan or pump flow where a failure must be detected for device control or safety shutdown.

Device Support: (all firmware revisions)

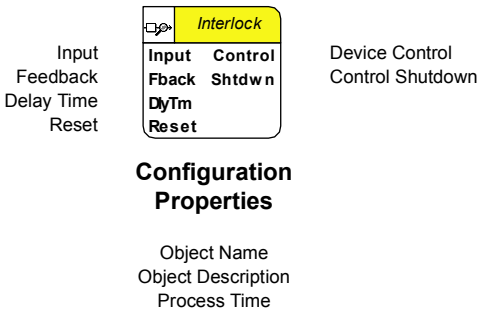
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 12 bytes
RAM: 28 bytes (standard controllers)
16 bytes (MN 800)

WP Tech Representation



Properties

Table–5.149 Interlock Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.150 Interlock Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Digital - The command signal for the interlock process. An OFF-to-ON transition starts the interlock verification sequence where the Feedback input is monitored for an actual verification.	—	A not active (NA) causes the Control output to be set to NA and the Shutdown output to be held OFF.
Fback	Feedback	Class: Digital - The feedback signal for the interlock process. This signal is continuously monitored and used to verify the command signal.	—	A not active (NA) causes all outputs to be held OFF.
DlyTm	Delay Time	Class: Analog - The verification delay time during the normal interlock start sequence.	0.0 to 10,000 seconds	A not active (NA) or negative value is evaluated as 0.0.
Reset	Reset	Class: Digital - Used to reset a verified interlock failure condition. An OFF-to-ON transition is manual reset. A not active (NA) causes the object to operate in an automatic reset mode during a verified interlock failure condition.	—	

Table–5.151 Interlock Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Control	Device Control	Class: Digital - This output typically follows the commanded input under normal interlock sequence conditions. This output is forced to OFF whenever a verified interlock failure condition occurs. This output is set to not active (NA) whenever the Input is NA.	OFF (0.0) ON (100.0)
Shtdwn	Control Shutdown	Class: Digital - This output is used to force shutdown of the control process if the Feedback input is not detected before the verification Delay Time expires or if a verification failure occurred during a normal interlock sequence. This output is forced to OFF whenever a verified interlock failure occurs.	OFF (0.0) ON (100.0)

Applying the Object

The Interlock object provides the logic needed for verifying a commanded ON signal against an independent digital logic feedback signal to determine the validity of a control state. The feedback signal must match the ON command within the interlock delay time (in seconds) otherwise an interlock failure is indicated at the object outputs.

Typical applications include proof of fan or pump flow where a failure must be detected for device control or safety shutdown. Under normal conditions, the Control Shutdown output remains ON. If a verified failure occurs, this output goes OFF until the Interlock object is manually or automatically reset.

Interlock Sequences Explained

Normal Interlock Start Sequence

An OFF-to-ON transition at the Input causes the Device Control output to immediately switch to ON. If the monitored signal at the Feedback input does not receive an ON signal within the assigned interlock delay time, both outputs (Device Control and Control Shutdown) are held OFF.
Reset of the shutdown condition can be setup as manual or automatic.
Successful verification of signals causes the Interlock object to begin monitoring the Input and Feedback signals as described next (normal interlock run sequence).

Normal Interlock Run Sequence

The Interlock object continues to monitor the Input and Feedback signals to determine if the control state fails during an actual run condition. If the Feedback signal does not match the Input ON control command for four seconds, both outputs (Device Control and Control Shutdown) are held OFF.
Reset of the shutdown condition can be setup as manual or automatic.

Reset from a Shutdown Condition

A control shutdown results from an interlock verification failure during either a normal interlock start or normal interlock run sequence. Depending on the state of the Reset input, a reset from a shutdown condition is automatic or manual.

- An NA at the Reset input results in an automatic reset routine. The Control Shutdown output is automatically reset (from OFF to ON) whenever the commanded Input is returned to OFF. The Interlock object is now ready to begin a normal interlock start sequence.
- A manual reset is performed by applying an OFF-to-ON transition to the Reset input. The Control Shutdown output is then returned to ON, and the Interlock object is now ready to begin a normal interlock start sequence.

Operation Following a Power Reset

Following a power reset, the Interlock object holds the Control Shutdown output in the shutdown state (OFF) until both the Input and Feedback signals are valid. The interlock algorithm then releases the Control Shutdown output (ON) and allows normal interlock verification to occur.

Operation if Input or Feedback Goes to Not Active (NA)

The interlock algorithm immediately sets the Control Shutdown output in the shutdown state (OFF) whenever either the Input or Feedback signals are found to be not active (NA), as the interlock algorithm does not have valid signals necessary to perform the interlock verification sequence.

In addition:

- An Input of not active (NA) sets the Device Control output to NA.
- A Feedback of not active (NA) sets the Device Control output to OFF.
- A simultaneous not active (NA) on both Input and Feedback sets the Device Control output to NA.

Timing Diagrams

Figure–5.72 below illustrates the operation of the Interlock object during normal interlock start sequences and a run sequence.

Normal Interlock Start Sequence

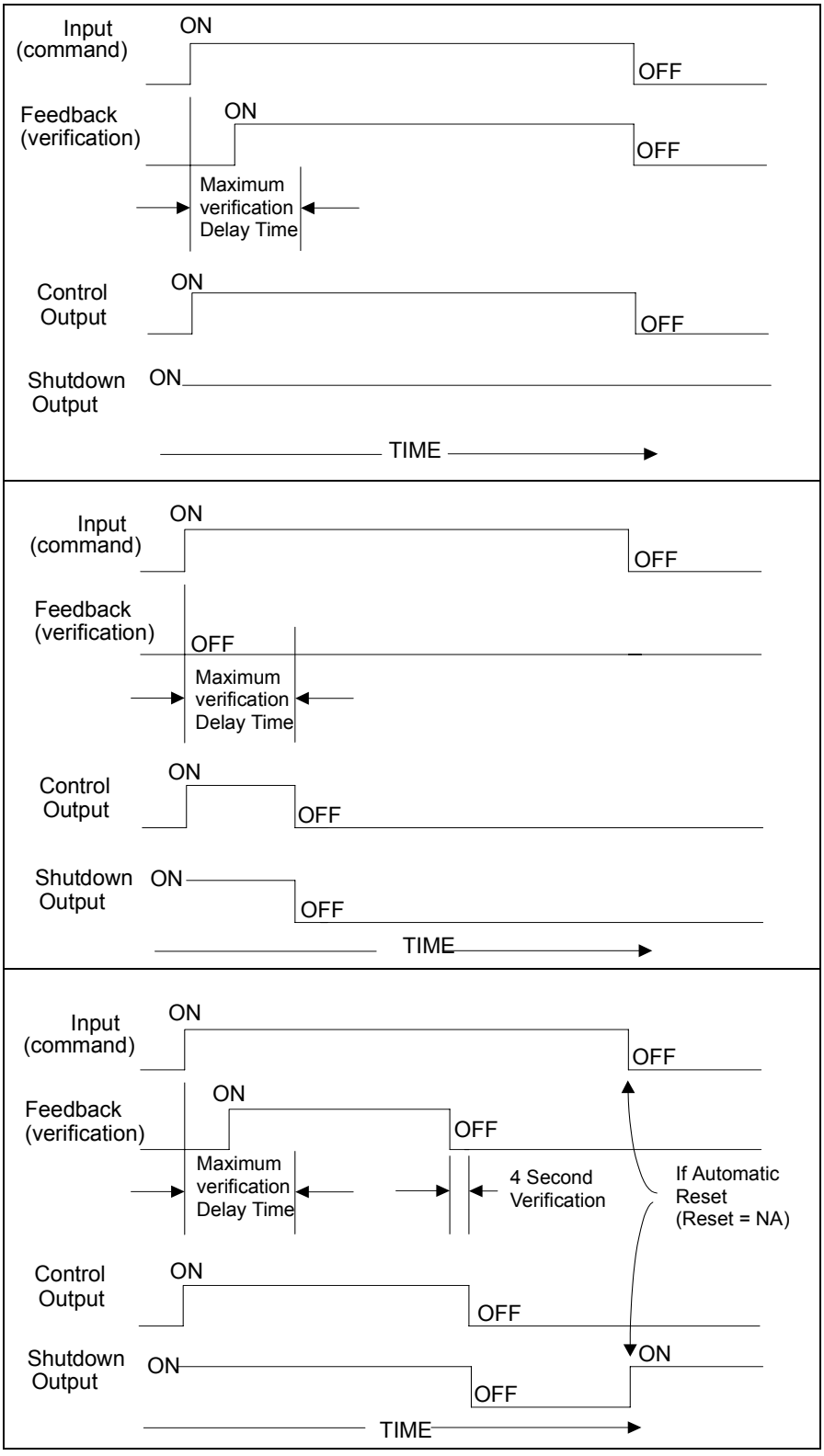
No Interlock Verification Failure

Normal Interlock Start Sequence,

With Interlock Verification Failure

Normal Interlock Run Sequence,

Verification Failure Detected During Run



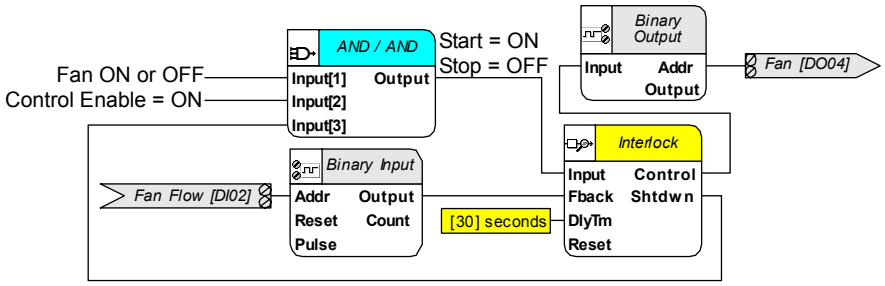
Figure–5.72 Normal Interlock Start and Run Sequences Compared.

Operation when Input (Command) is OFF

Changes at the Feedback input are ignored while the Input is OFF. This can occur if a user has manually overridden the control application, for instance, by turning on a piece of equipment using a local Hand/Off/Auto switch. In this case, the verification sequence is not performed.

Example

An Interlock object is used in the example application below Figure-5.73 for proof of air flow. The Input is connected to the commanded fan start / stop signal (ON or OFF), and the Output directly follows the Input state (command).



Figure–5.73 Example Interlock Object Used in a Fan Proof of Flow Application.

The Control Shutdown output always remains ON when the interlock algorithm determines that the control state is normal. During a start sequence, if the monitored flow at the Feedback input is not detected for any time period exceeding the (verification) Delay Time, the Control Shutdown output goes OFF, signaling a shutdown condition. In this example, Delay Time is assigned a constant value of 30 seconds. As is typically done, the Control Shutdown output is logic ANDed with a control enable signal to safely disable fan control if proof of flow failure is detected.

In this example, reset of a control shutdown sequence is automatic, as the Reset input of the Interlock object is left unconnected (not active or NA).

Interstage Delay (3)

Object Usage: The Interstage Delay (3) object provides both interstage delay timer and dual minimum functions while maintaining the interstage sequencing order. This object is typically paired with a Sequence (3) object to provide staggered start / stop linear sequence control with short cycling protection at each output stage.

The Interstage Delay (3) object works like the Interstage Delay (6) and Interstage Delay (10) objects, except with fewer output stages.

Device Support: (all firmware revisions)

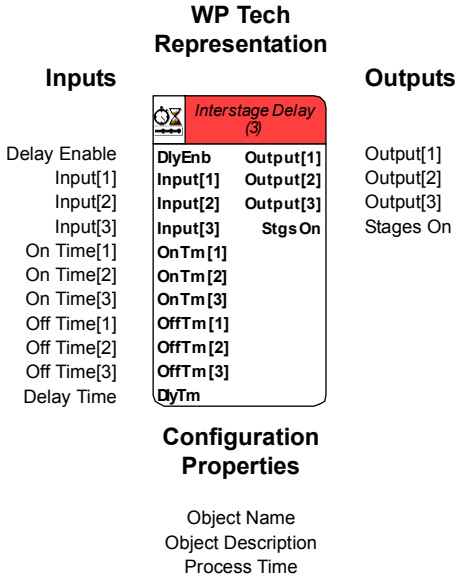
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 26 bytes
RAM: 44 bytes (standard controllers)
18 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Timer and Sequence Control

Properties

Table–5.152 Interstage Delay (3) Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.153 Interstage Delay (3) Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
DlyEnb	Delay Enable	Class: Digital - An ON or not active (NA) enables the Interstage Delay and dual-minimum timeout algorithm. An OFF disables any time delays.	—	
Input[1]	Input [1]	Class: Digital - First input in the interstage sequence that controls Output[1].	—	If any Input is not active (NA), the associated output and all outputs higher in the sequence are held at OFF. See the Sequence Table for more information.
Input[2]	Input [2]	Class: Digital - Second input in the interstage sequence that controls Output[2].	—	
Input[3]	Input [3]	Class: Digital - Third input in the interstage sequence that controls Output[3].	—	
OnTm[1]	On Time [1]	Class: Analog - The minimum on time (in minutes) associated with Output[1].	0.0 to 1,000 minutes	A negative value or not active (NA) is evaluated as 0.0 minutes (no Minimum On time).
OnTm[2]	On Time [2]	Class: Analog - The minimum on time (in minutes) associated with Output[2].	0.0 to 1,000 minutes	
OnTm[3]	On Time [3]	Class: Analog - The minimum on time (in minutes) associated with Output[3].	0.0 to 1,000 minutes	
OffTm[1]	Off Time [1]	Class: Analog - The minimum off time (in minutes) associated with Output[1].	0.0 to 1,000 minutes	A negative value or not active (NA) is evaluated as 0.0 minutes (no Minimum Off time).
OffTm[2]	Off Time [2]	Class: Analog - The minimum off time (in minutes) associated with Output[2].	0.0 to 1,000 minutes	
OffTm[3]	Off Time [3]	Class: Analog - The minimum off time (in minutes) associated with Output[3].	0.0 to 1,000 minutes	
DlyTm	Delay Time	Class: Analog - Defines the interstage delay time in seconds.	0.0 to 10,000 seconds	A NA or negative value is evaluated as 0.0 seconds

Table–5.154 Interstage Delay (3) Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital - The first or lowest output stage in the interstage sequence. This output typically follows the commanded Input[1] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[2]	Output [2]	Class: Digital - The second output stage in the interstage sequence. This output typically follows the commanded Input[2] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[3]	Output [3]	Class: Digital - The third output stage in the interstage sequence. This output typically follows the commanded Input[3] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
StgsOn	Stages On	Class: Analog - Reflects the number of output stages ON during the active sequence operation.	0, 1, 2, or 3

Applying the Object

The Interstage Delay (3) object provides both interstage and dual-minimum functions while maintaining an interstage sequencing order. This object is typically paired with the Sequence (3) object to provide a staggered start / stop linear sequence control with short cycle protection at each output stage.

The Interstage Delay object monitors the digital Inputs[1], [2], and [3], and determines the output configuration based upon input linearity, not active (NA) values, and out of sequence conditions. Input linearity is defined as a linear sequence from Input[1] to Input [3] where the first stage Input[1] is the first stage ON and the last stage OFF. The object adjusts the outputs (stage up / stage down) to follow the series of ON input requests. The output sequence is limited by the first OFF or NA found at the inputs.

An ON or not active (NA) at the Delay Enable input allows operation of all interstage delay and dual-minimum functions. If the Delay Enable receives an OFF, all interstage delay and dual-minimum timeouts are disabled. Upon system reset, the interstage delay and minimum off timers are initialized and activated to their assigned timeout values.

Interstage Sequence

Output[1] is set to ON upon Input[1] request, after the minimum off timeout for Output[1] has elapsed. Output[2] is set to ON upon Input [2] request after the assigned interstage delay and minimum off timeouts for Output[2] have elapsed. Output[1] remains ON as long as an Input[1] ON request remains.

Output[1] is reset to OFF from an Input[1] request provided the higher output in the sequence Output[2] is OFF, the interstage timeouts between all higher stages has elapsed, and the minimum on timeout for Output[1] is complete.

Request for Additional Stages

Output[n] is set to ON upon Input[n] request after the assigned interstage delay and minimum off timeouts for Output[n] have elapsed. The interstage delay is re-initialized in preparation for progression to the next stage [n+1]. Output[n] remains ON as long as an Input[n] ON request remains.

Output[n] is reset to OFF from an Input[n] OFF request provided that the higher output in the sequence Output[n+1] is OFF, the interstage timeouts between all higher stages has elapsed, and the minimum on timeout for Output[n] is complete.

Sequence Input Not Active (NA)

A not active (NA) at any input causes the associated output as well as all outputs higher in the sequence to be immediately set to OFF. Also, all associated interstage delays and minimum timeouts are canceled.

The following tables show the input-to-output results of both a normal interstage sequence Table–5.155 and one with out-of-sequence or not active (NA) input conditions Table–5.156.

Table–5.155 Interstage Delay (3) Object Sequence (Normal Sequence).

Input[1] Lowest	Input[2]	Input[3] Highest	Output[1] Lowest	Output[2]	Output[3] Highest
OFF	OFF	OFF	OFF	OFF	OFF
ON	OFF	OFF	ON	OFF	OFF
ON	ON	OFF	ON	ON	OFF
ON	ON	ON	ON	ON	ON
ON	ON	OFF	ON	ON	OFF
ON	OFF	OFF	ON	OFF	OFF
OFF	OFF	OFF	OFF	OFF	OFF

Note: Output configurations shown are after all associated interstage delay and minimum timeouts have expired.

Table–5.156 Interstage Delay (3) Object Sequence (Out-of-Sequence or NA).

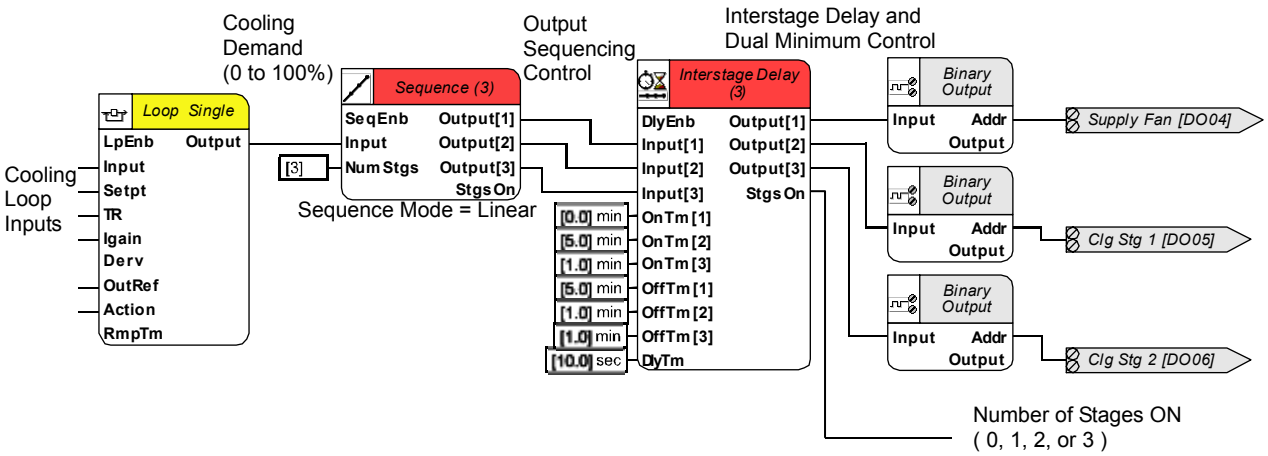
Input[1] Lowest	Input[2]	Input[3] Highest	Output[1] Lowest	Output[2]	Output[3] Highest
OFF	ON	ON	OFF	OFF	OFF
ON	OFF	ON	ON	OFF	OFF
NA	ON	OFF	OFF	OFF	OFF
ON	NA	ON	ON	OFF	OFF
ON	ON	NA	ON	ON	OFF

Note: Not active (NA) inputs cause outputs to respond immediately, regardless of any associated delay or minimum timeouts.

Example

Figure–5.74 shows an Interstage Delay (3) object used to provide the control sequence and time delays for a cooling application that has a supply fan and two stages of DX cooling.

As the cooling demand signal increases from 0 to 100%, the Sequence (3) object provides a three staged linear output causing the first stage (fan) to energize. As demand continues to increase, first stage cooling (DX compressor) is energized (a minimum of 10 seconds interstage delay must occur before this happens). Once the first stage of DX cooling is energized, the compressor must remain on for at least 5 minutes before being allowed to turn off as cooling demand decreases.



Figure–5.74 Example Interstage Delay (3) Object in a Cooling Application.

As demand continues to increase and reaches 100%, the second stage cooling will be energized (again, a minimum of 10 seconds interstage delay must occur first). Once the second stage cooling is energized, the compressor must remain on for at least 1 minute before being allowed to turn off as cooling demand decreases.

As cooling demand decreases, each stage is sequentially de-energized utilizing the appropriate interstage and minimum on timeouts. All stages must complete their associated minimum off timeouts before being allowed to energize for another cooling cycle. The Stages On output reflects the actual number of stages ON during the active sequence operation.

Interstage Delay (6)

Object Usage: The Interstage Delay (6) object provides both interstage delay timer and dual-minimum functions while maintaining the interstage sequencing order. This object is typically paired with a Sequence (6) object to provide staggered start / stop linear sequence control with short cycling protection at each output stage.

The Interstage Delay (6) object works like the Interstage Delay (3) and Interstage Delay (10) objects, except it has six output stages.

Device Support: (all firmware revisions)

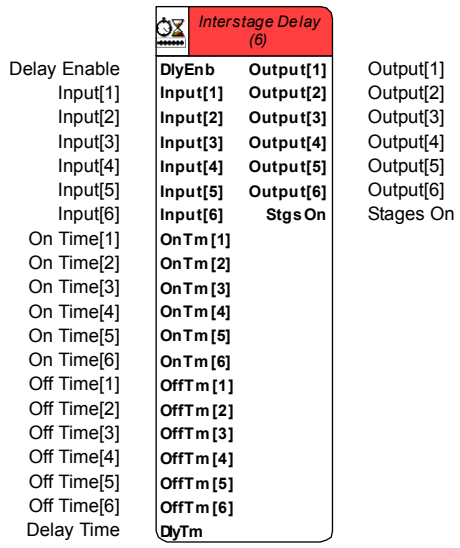
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 44 bytes
RAM: 74 bytes (standard controllers)
30 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Timer and Sequence Control

Properties

Table–5.157 Interstage Delay (6) Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.158 Interstage Delay (6) Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
DlyEnb	Delay Enable	Class: Digital - An ON or not active (NA) enables the Interstage Delay and dual-minimum timeout algorithm. An OFF disables any time delays.	—	

Table–5.158 Interstage Delay (6) Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Digital - First input in the interstage sequence that controls Output[1].	—	If any Input is not active (NA), the associated output and all outputs higher in the sequence are held at OFF. See the Sequence Table for more information.
Input[2]	Input [2]	Class: Digital - Second input in the interstage sequence that controls Output[2].	—	
⋮	⋮	⋮	⋮	
Input[6]	Input [6]	Class: Digital - Sixth input in the interstage sequence that controls Output[6].	—	
OnTm[1]	On Time [1]	Class: Analog - The minimum on time (in minutes) associated with Output[1].	0.0 to 1,000 minutes	A negative value or not active (NA) is evaluated as 0.0 minutes (no Minimum On time).
OnTm[2]	On Time [2]	Class: Analog - The minimum on time (in minutes) associated with Output[2].	0.0 to 1,000 minutes	
⋮	⋮	⋮	⋮	
OnTm[6	On Time [6]	Class: Analog - The minimum on time (in minutes) associated with Output[6].	0.0 to 1,000 minutes	
OffTm[1]	Off Time [1]	Class: Analog - The minimum off time (in minutes) associated with Output[1].	0.0 to 1,000 minutes	A negative value or not active (NA) is evaluated as 0.0 minutes (no Minimum Off time).
OffTm[2]	Off Time [2]	Class: Analog - The minimum off time (in minutes) associated with Output[2].	0.0 to 1,000 minutes	
⋮	⋮	⋮	⋮	
OffTm[6]	Off Time [6]	Class: Analog - The minimum off time (in minutes) associated with Output[6].	0.0 to 1,000 minutes	
DlyTm	Delay Time	Class: Analog - Defines the interstage delay time in seconds.	0.0 to 10,000 seconds	A NA or negative value is evaluated as 0.0 seconds

Table–5.159 Interstage Delay (6) Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital - The first or lowest output stage in the interstage sequence. This output typically follows the commanded Input[1] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[2]	Output [2]	Class: Digital - The second output stage in the interstage sequence. This output typically follows the commanded Input[2] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[3]	Output [3]	Class: Digital - The third output stage in the interstage sequence. This output typically follows the commanded Input[3] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[4]	Output [4]	Class: Digital - The fourth output stage in the interstage sequence. This output typically follows the commanded Input[4] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[5]	Output [5]	Class: Digital - The fifth output stage in the interstage sequence. This output typically follows the commanded Input[5] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)

Table–5.159 Interstage Delay (6) Object Output Properties. (Continued)

Abbrev.	Name	Class / Description	Valid Values
Output[6]	Output [6]	Class: Digital - The sixth output stage in the interstage sequence. This output typically follows the commanded Input[6] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
StgsOn	Stages On	Class: Analog - Reflects the number of output stages ON during the active sequence operation.	0, 1, 2, 3, 4, 5, or 6

Applying the Object

The Interstage Delay (6) object provides both interstage and dual-minimum functions while maintaining an interstage sequencing order. This object is typically paired with a Sequence (6) object to provide a staggered start / stop linear sequence control with short cycle protection at each output stage.

The Interstage Delay object monitors the digital Inputs[1], [2], [3], [4], [5], and [6] and determines the output configuration based upon input linearity, not active (NA) values, and out of sequence conditions. Input linearity is defined as a linear sequence from Input[1] to Input [6] where the first stage Input[1] is the first stage ON and the last stage OFF. The object adjusts the outputs (stage up / stage down) to follow the series of ON input requests. The output sequence is limited by the first OFF or NA found at the inputs.

An ON or not active (NA) at the Delay Enable input allows operation of all interstage delay and dual-minimum functions. If the Delay Enable receives an OFF, all interstage delay and dual-minimum timeouts are disabled. Upon system reset, the interstage delay and minimum off timers are initialized and activated to their assigned timeout values.

Interstage Sequence

Output[1] is set to ON upon Input[1] request, after the minimum off timeout for Output[1] has elapsed. Output[2] is set to ON upon Input [2] request after the assigned interstage delay and minimum off timeouts for Output[2] have elapsed. Output[1] remains ON as long as an Input[1] ON request remains.

Output[1] is reset to OFF from an Input[1] request provided the higher output in the sequence Output[2] is OFF, the interstage timeouts between all higher stages has elapsed, and the minimum on timeout for Output[1] is complete.

Request for Additional Stages

Output[n] is set to ON upon Input[n] request after the assigned interstage delay and minimum off timeouts for Output[n] have elapsed. The interstage delay is re-initialized in preparation for progression to the next stage [n+1]. Output[n] remains ON as long as an Input[n] ON request remains.

Output[n] is reset to OFF from an Input[n] OFF request provided that the higher output in the sequence Output[n+1] is OFF, the interstage timeouts between all higher stages has elapsed, and the minimum on timeout for Output[n] is complete.

Sequence Input Not Active (NA)

A not active (NA) at any input causes the associated output as well as all outputs higher in the sequence to be immediately set to OFF. Also, all associated interstage delays and minimum timeouts are canceled.

The following tables show the input-to-output results of both a normal interstage sequence Table–5.160 and out-of-sequence or not active (NA) input conditions Table–5.161.

Table–5.160 Interstage Delay (6) Object Sequence (Normal Sequence).

In[1] Lowest	In[2]	In[3]	In[4]	In[5]	In[6] Highest	Out[1] Lowest	Out[2]	Out[3]	Out[4]	Out[5]	Out[6]
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	OFF	OFF	OFF	OFF	OFF	ON	OFF	OFF	OFF	OFF	OFF
ON	ON	OFF	OFF	OFF	OFF	ON	ON	OFF	OFF	OFF	OFF
ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF
ON	ON	ON	ON	OFF	OFF	ON	ON	ON	ON	OFF	OFF
ON	ON	ON	ON	ON	OFF	ON	ON	ON	ON	ON	OFF
ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON
ON	ON	ON	ON	ON	OFF	ON	ON	ON	ON	ON	OFF
ON	ON	ON	ON	OFF	OFF	ON	ON	ON	ON	OFF	OFF
ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF
ON	ON	OFF	OFF	OFF	OFF	ON	ON	OFF	OFF	OFF	OFF
ON	OFF	OFF	OFF	OFF	OFF	ON	OFF	OFF	OFF	OFF	OFF
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

Note: Output configurations shown are after all associated interstage delay and minimum timeouts have expired.

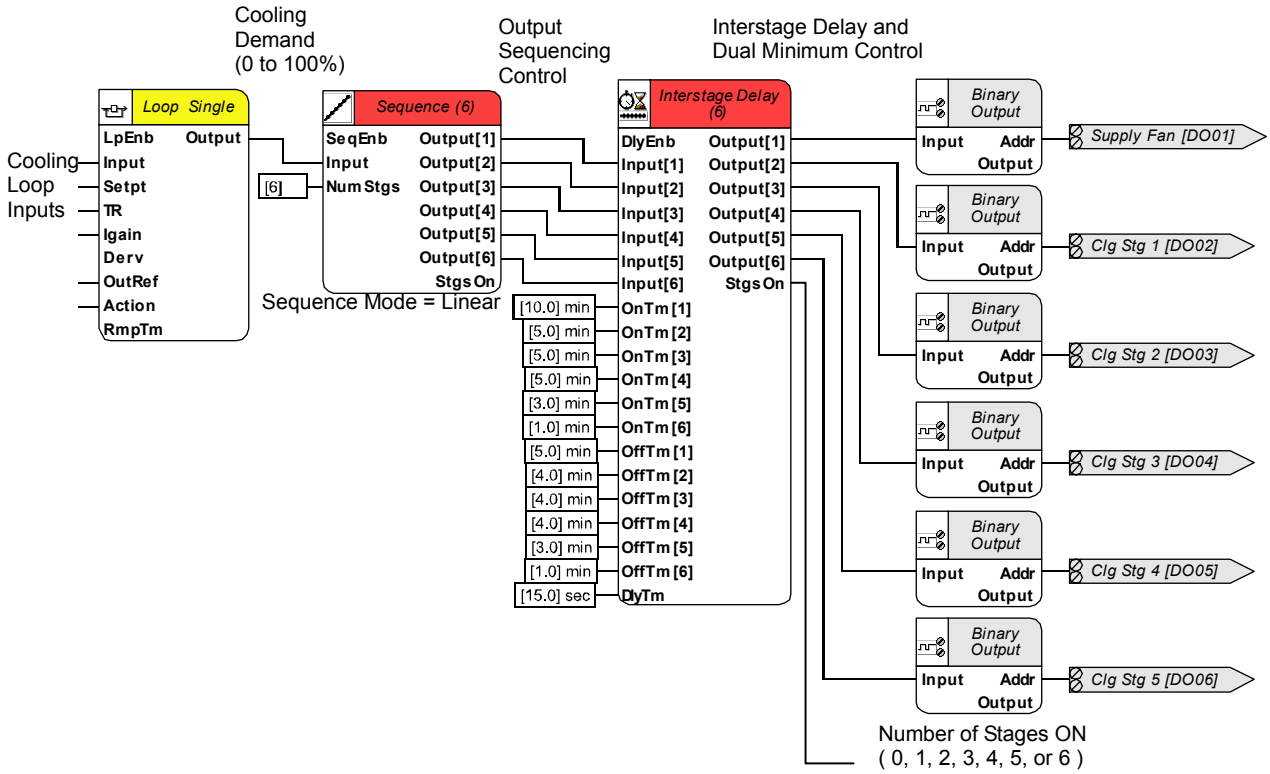
Table–5.161 Example Results of Not Active (NA) and Out of Sequence Inputs for an Interstage Delay (6) Object.

In[1] Lowest	In[2]	In[3]	In[4]	In[5]	In[6] Highest	Out[1] Lowest	Out[2]	Out[3]	Out[4]	Out[5]	Out[6]
OFF	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	OFF	ON	ON	OFF	ON	ON	OFF	OFF	OFF	OFF
NA	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	NA	ON	ON	OFF	ON	ON	OFF	OFF	OFF	OFF
ON	ON	ON	ON	ON	NA	ON	ON	ON	ON	ON	OFF

Note: Not active (NA) inputs cause outputs to respond immediately, regardless of any associated delay or minimum timeouts.

Example

Figure–5.75 shows an Interstage Delay (6) object used to provide the control sequence and time delays for a cooling application with a supply fan and five stages of DX cooling.



Figure–5.75 Example Interstage Delay (6) Object in a Cooling Application.

As the cooling demand signal increases from 0 to 100%, the Sequence (6) object provides a six staged linear output causing the first stage (fan) to energize. As demand continues to increase, first stage cooling (DX compressor) is energized (a minimum of 15 seconds interstage delay must occur before this happens). Once the first stage of DX cooling is energized, the compressor must remain on for at least 5 minutes before being allowed to turn off as cooling demand decreases.

As demand continues to increase additional stages will be energized (again, a minimum of 15 seconds interstage delay must occur between each successive stage). At a cooling demand of 100%, all six stages will be energized. With the sixth stage of cooling energized, the compressor must remain on for at least 1 minute before being allowed to turn off as cooling demand decreases.

As cooling demand decreases, each stage is sequentially de-energized utilizing the appropriate interstage and minimum on timeouts. All stages must complete their associated minimum off timeouts before being allowed to energize for another cooling cycle. The Stages On output reflects the actual number of stages ON during the active sequence operation.

Interstage Delay (10)

Object Usage: The Interstage Delay (10) object provides both interstage delay timer and dual-minimum functions while maintaining the interstage sequencing order. This object is typically paired with a Sequence (10) object to provide staggered start / stop linear sequence control with short cycling protection at each output stage.

The Interstage Delay (10) object works like the Interstage Delay (3) and Interstage Delay (6) objects, but provides the most (10) output stages.

Device Support: (all firmware revisions)

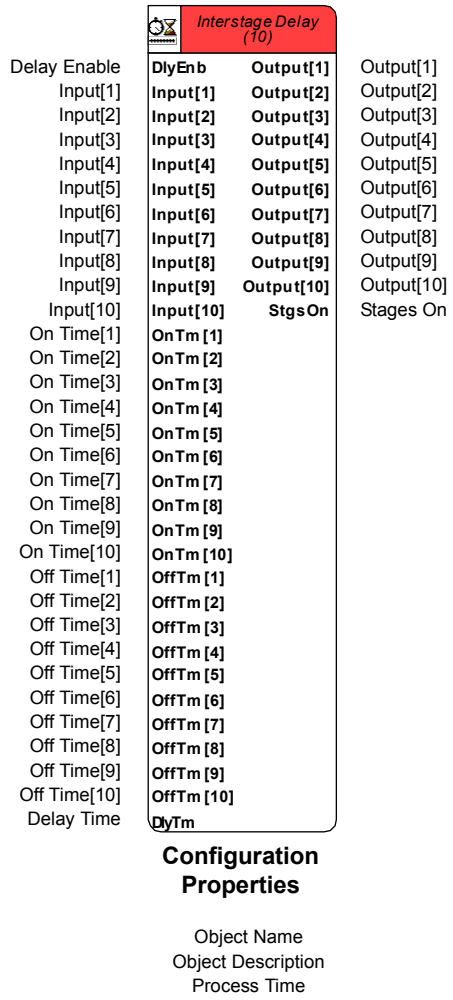
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 68 bytes
RAM: 114 bytes (standard controllers)
46 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Timer and Sequence Control

Properties

Table–5.162 Interstage Delay (10) Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.

Table–5.162 Interstage Delay (10) Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.163 Interstage Delay (10) Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
DlyEnb	Delay Enable	Class: Digital - An ON or not active (NA) enables the Interstage Delay and dual-minimum timeout algorithm. An OFF disables any time delays.	—	
Input[1]	Input [1]	Class: Digital - First input in the interstage sequence that controls Output[1].	—	If any Input is not active (NA), the associated output and all outputs higher in the sequence are held at OFF. See the Sequence Table for more information.
Input[2]	Input [2]	Class: Digital - Second input in the interstage sequence that controls Output[2].	—	
⋮	⋮	⋮	⋮	
Input[10]	Input [10]	Class: Digital - Tenth input in the interstage sequence that controls Output[10].	—	
OnTm[1]	On Time [1]	Class: Analog - The minimum on time (in minutes) associated with Output[1].	0.0 to 1,000 minutes	A negative value or not active (NA) is evaluated as 0.0 minutes (no Minimum On time).
OnTm[2]	On Time [2]	Class: Analog - The minimum on time (in minutes) associated with Output[2].	0.0 to 1,000 minutes	
⋮	⋮	⋮	⋮	
OnTm[10]	On Time [10]	Class: Analog - The minimum on time (in minutes) associated with Output[10].	0.0 to 1,000 minutes	
OffTm[1]	Off Time [1]	Class: Analog - The minimum off time (in minutes) associated with Output[1].	0.0 to 1,000 minutes	A negative value or not active (NA) is evaluated as 0.0 minutes (no Minimum Off time).
OffTm[2]	Off Time [2]	Class: Analog - The minimum off time (in minutes) associated with Output[2].	0.0 to 1,000 minutes	
⋮	⋮	⋮	⋮	
OffTm[10]	Off Time [10]	Class: Analog - The minimum off time (in minutes) associated with Output[10].	0.0 to 1,000 minutes	
DlyTm	Delay Time	Class: Analog Defines the interstage delay time in seconds.	0.0 to 10,000 seconds	A NA or negative value is evaluated as 0.0 seconds

Table–5.164 Interstage Delay (10) Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital - The first or lowest output stage in the interstage sequence. This output typically follows the commanded Input[1] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)

Table–5.164 Interstage Delay (10) Object Output Properties. (Continued)

Abbrev.	Name	Class / Description	Valid Values
Output[2]	Output [2]	Class: Digital - The second output stage in the interstage sequence. This output typically follows the commanded Input[2] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[3]	Output [3]	Class: Digital - The third output stage in the interstage sequence. This output typically follows the commanded Input[3] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[4]	Output [4]	Class: Digital - The fourth output stage in the interstage sequence. This output typically follows the commanded Input[4] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[5]	Output [5]	Class: Digital - The fifth output stage in the interstage sequence. This output typically follows the commanded Input[5] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[6]	Output [6]	Class: Digital - The sixth output stage in the interstage sequence. This output typically follows the commanded Input[6] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[7]	Output [7]	Class: Digital - The seventh output stage in the interstage sequence. This output typically follows the commanded Input[7] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[8]	Output [8]	Class: Digital - The eighth output stage in the interstage sequence. This output typically follows the commanded Input[8] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[9]	Output [9]	Class: Digital - The ninth output stage in the interstage sequence. This output typically follows the commanded Input[9] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
Output[10]	Output [10]	Class: Digital - The tenth output stage in the interstage sequence. This output typically follows the commanded Input[10] under normal interstage delay sequence and dual-minimum timeout conditions.	OFF (0.0) ON (100.0)
StgsOn	Stages On	Class: Analog Reflects the number of output stages ON during the active sequence.	0 through 10 (integer)

Applying the Object

The Interstage Delay (10) object provides both interstage and dual-minimum functions while maintaining an interstage sequencing order. This object is typically paired with the Sequence (10) object to provide a staggered start / stop linear sequence control with short cycle protection at each output stage.

The Interstage Delay object monitors the digital Inputs[1] through [10] and determines the output configuration based upon input linearity, not active (NA) values, and out of sequence conditions. Input linearity is defined as a linear sequence from Input[1] to Input[10] where the first stage Input[1] is the first stage ON and the last stage OFF. The object adjusts the outputs (stage up / stage down) to follow the series of ON input requests. The output sequence is limited by the first OFF or NA found at the inputs.

An ON or not active (NA) at the Delay Enable input allows operation of all interstage delay and dual-minimum functions. If the Delay Enable receives an OFF, all interstage delay and dual-minimum timeouts are disabled. Upon system reset, the interstage delay and minimum off timers are initialized and activated to their assigned timeout values.

Interstage Sequence

Output[1] is set to ON upon Input[1] request, after the minimum off timeout for Output[1] has elapsed. Output[2] is set to ON upon Input [2] request after the assigned interstage delay and minimum off timeouts for Output[2] have elapsed. Output[1] remains ON as long as an Input[1] ON request remains.

Output[1] is reset to OFF from an Input[1] request provided the higher output in the sequence Output[2] is OFF, the interstage timeouts between all higher stages has elapsed, and the minimum on timeout for Output[1] is complete.

Request for Additional Stages

Output[n] is set to ON upon Input[n] request after the assigned interstage delay and minimum off timeouts for Output[n] have elapsed. The interstage delay is re-initialized in preparation for progression to the next stage [n+1]. Output[n] remains ON as long as an Input[n] ON request remains.

Output[n] is reset to OFF from an Input[n] OFF request provided that the higher output in the sequence Output[n+1] is OFF, the interstage timeouts between all higher stages has elapsed, and the minimum on timeout for Output[n] is complete.

Sequence Input Not Active (NA)

A not active (NA) at any input causes the associated output as well as all outputs higher in the sequence to be immediately set to OFF. Also, all associated interstage delays and minimum timeouts are canceled.

The following tables show the input-to-output results of both a normal interstage sequence Figure-5.165 and ones with out-of-sequence or not active (NA) input conditions Figure-5.166.

Table–5.165 Interstage Delay (10) Object Sequence (Normal Sequence).

In [1] Low	In [2]	In [3]	In [4]	In [5]	In [6]	In [7]	In [8]	In [9]	In [10] High	Out [1] Low	Out [2]	Out [3]	Out[4]	Out [5]	Out [6]	Out [7]	Out. [[8]	Out [9]	Out [10] High
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF
ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF
ON	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF
ON	ON	ON	ON	ON	ON	ON	ON	OFF	OFF	ON	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF	ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF
ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON	ON
ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF	ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF
ON	ON	ON	ON	ON	ON	ON	ON	OFF	OFF	ON	ON	ON	ON	ON	ON	ON	ON	OFF	OFF
ON	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	ON	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF
ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF
ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF
ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

Note: Output configurations shown are after all associated interstage delay and minimum timeouts have expired.

Table–5.166 Example Results of Not Active (NA) and Out of Sequence Inputs for an Interstage Delay (10) Object.

In [1] Low	In [2]	In [3]	In [4]	In [5]	In [6]	In [7]	In [8]	In [9]	In [10] High	Out [1] Low	Out [2]	Out [3]	Out[4]	Out [5]	Out [6]	Out [7]	Out. [[8]	Out [9]	Out [10] High
OFF	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	ON	OFF	ON	ON	OFF	OFF	OFF	OFF	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF
NA	ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	ON	ON	NA	ON	ON	OFF	OFF	OFF	ON	ON	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF
ON	ON	ON	ON	ON	ON	ON	ON	ON	NA	ON	ON	ON	ON	ON	ON	ON	ON	ON	OFF

Note: Not active (NA) inputs cause outputs to respond immediately, regardless of any associated delay or minimum timeouts.

Example

The Interstage Delay (10) object functions exactly like the other Interstage Delay (6) and Interstage Delay (3) objects, except it features 10 sequence inputs and outputs and is typically paired with a Sequence (10) object.

See the previous examples for the Interstage Delay (6) object and Interstage Delay (3) object for typical application examples.

Latch

Object Usage: The Latch object samples two input signals and provides two different functions based on how the inputs are connected. The functions are:

- Standard Digital Latch
- Sample and Hold

Used as a digital latch, the Output remains in an ON state following the first OFF-to-ON transition at the Latch input. Used as a sample and hold, the Output remains at the analog value sampled at the Data input following the first OFF-to-ON transition at the Latch input. A Reset input provides a method of clearing a latched ON or held analog value.

Device Support: (all firmware revisions)

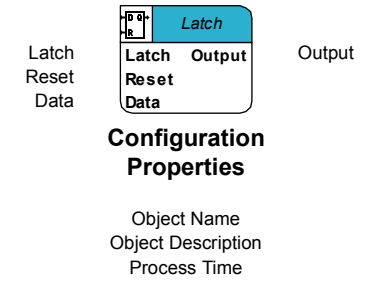
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 14 bytes (standard controllers)
4 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Digital Logic Objects

Object Name	Digital Object Algorithm (all are three-input unless noted)
AND / AND	In1 AND In2 AND In3
AND / OR	(In1 AND In2) OR In3
Clocked SR	Clocked Set-Reset Flip-Flop Logic
EXOR	Two-input, Exclusive OR
Latch	Digital Sample and Hold or Latch
OR / AND	(In1 OR In2) AND In3
OR / OR	In1 OR In2 OR In3
SR Flip-Flop	Two-input, Set-Reset Flip-Flop Logic

Properties

Table–5.167 Latch Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.168 Latch Object Input Properties

Abbrev.	Name	Class / Description	Range / Selection	Notes
Latch	Latch	Class: Digital - As a latch, an OFF-to-ON transition latches the output. As a sample and hold, an OFF-to-ON transition triggers the sample hold function.	OFF (0) ON (100)	See the Input / Output Validity Tables and Timing Diagrams for both the Latch Function and Sample and Hold Function for further details on each input property, including how a not active (NA) is processed.
Reset	Reset	Class: Digital - As a latch, an OFF-to-ON transition is used to clear the output from the last digital latch. As a sample and hold, OFF-to-ON transition releases the output value which allows the output value to track the value at the Data input.	OFF (0) ON (100)	
Data	Data	Class: Digital / Analog - As a latch, this input is externally connected to the same source used by the Latch input. As a sample and hold, this input is the data value to be sampled.	Digital: OFF (0.0) ON (100.0) Analog: -163.83 to 16383	

Table–5.169 Latch Object Output Properties

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital / Analog - The result of the last digital Latch (Digital) or data sample hold (Analog) received at the Data input. See the Input / Output Validity Tables and Timing Diagrams for both the Latch Function and Sample and Hold Function for further details.	Digital: OFF (0.0) ON (100.0) Analog: -163.83 to 16383

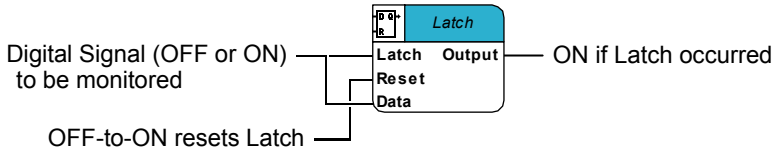
Applying the Object

The Latch object works in the following two configurations:

- A Digital Latch to capture an OFF-to-ON transition at the Data input.
- An analog “Sample and Hold” to capture an analog value at the Data input when an OFF-to-ON transition occurs at the digital Latch input.

Digital Latch

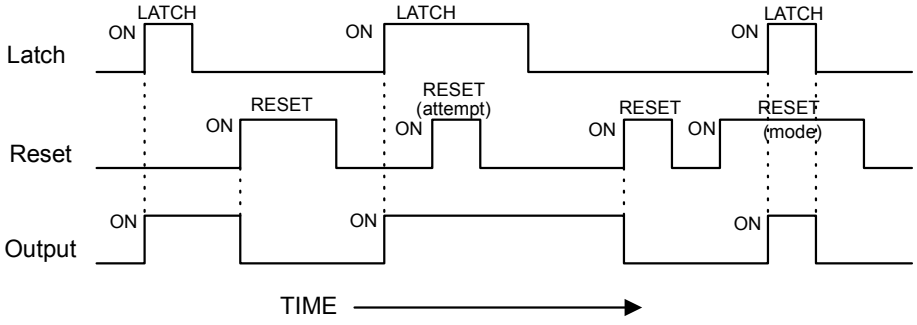
The Digital Latch configuration requires the Latch and Data inputs both connected to the same digital signal source Figure-5.76.



Figure–5.76 Latch Object Configured for Digital Latch Operation.

The output of the digital latch function is set to ON whenever Latch / Data is set to Digital ON. The output remains latched in this ON state until a reset signal occurs on the Reset input (provided that the Latch / Data input signal has previously returned to an OFF state). Holding the reset signal (Reset) at a Digital ON causes the output to directly track the digital conditions present at Data.

The following timing diagram illustrates the operation of the Latch object in the digital Latch configuration.



Figure–5.77 Digital Latch Timing Diagram for Latch Object.

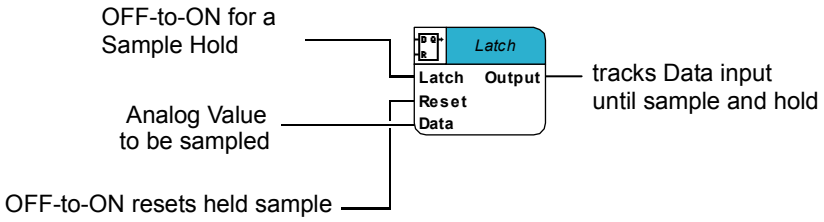
Not active (NA) inputs affect the Output results as shown in the following chart Figure-5.170.

Table–5.170 Digital Latch Operation with Not Active (NA) Input(s).

Latch	Reset	Data	Output
Valid	Valid	Valid	Normal operation Figure–5.77
Valid	NA	Valid	Single trigger latch
NA	Valid	NA	Output set to NA
NA	NA	NA	Output set to NA

Sample and Hold

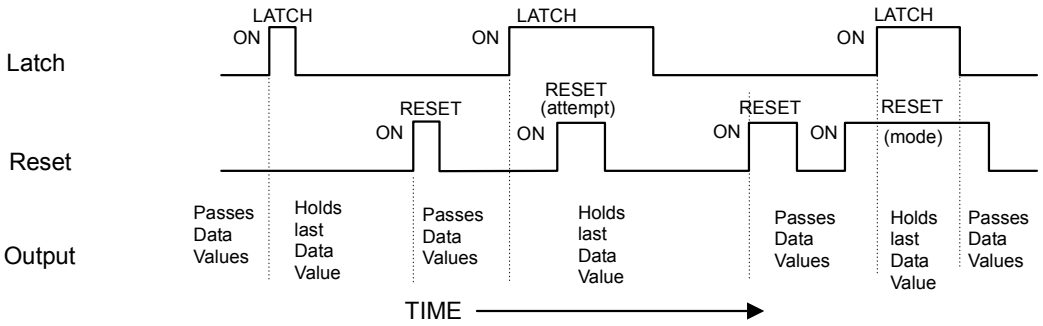
The Sample and Hold configuration of the Latch object is achieved with an analog value connected to the Data input. A digital OFF-to-ON transition at the Latch input triggers a data sample, held at the Output.



Figure–5.78 Latch Object Configured for Sample and Hold Operation.

The Output holds the analog value found on Data during the latch until a reset signal occurs on the Reset input (providing the Latch input has previously returned to an OFF state). Holding the reset signal (Reset) at ON causes the output to directly track the signal condition present at Data as long as Latch remains in the OFF state.

The following timing diagram illustrates the operation of the Latch object in the Sample and Hold configuration.



Figure–5.79 Sample and Hold Timing Diagram for the Latch Object.

Not active (NA) inputs affect the Output results as shown in Table–5.171.

Table–5.171 Sample and Hold Operation with Not Active (NA) Input(s).

Latch	Reset	Data	Output
Valid	Valid	Valid	Normal sample and hold operation.
NA	Valid	Valid	Tracks Data value.
Valid	NA	Valid	Single sample and hold. Tracks data value until ON latch occurs
NA	NA	Valid	Tracks Data value.
(don't care)	(don't care)	NA	Output set to NA.

Limit

Object Usage: The Limit object provides a method of restricting the range of the analog value received at the Input to an Output value between the values present at the Output Minimum and Output Maximum inputs. Input values within this range pass directly to the Output, while Input values outside this range produce the corresponding minimum or maximum limit value at the Output. Typical use is to limit an output of a loop or to limit the range of setpoint values in an application.

Device Support: (all firmware revisions)

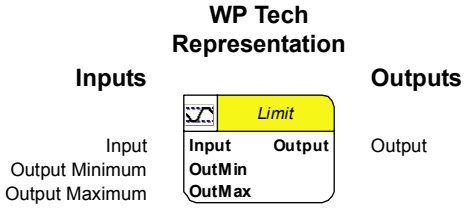
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.172 Limit Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.173 Limit Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Analog - The input signal to be limited by the object. A not active (NA) is passed directly to the Output.	-163.83 to 16383	

Table–5.173 Limit Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
OutMin	Output Minimum	Class: Analog - The assigned low limit or minimum output value. A not active (NA) disables the low limit function within the object.	-163.83 to 16383	
OutMax	Output Maximum	Class: Analog - The assigned high limit or maximum output value. A not active (NA) disables the high limit function within the object.	-163.83 to 16383	

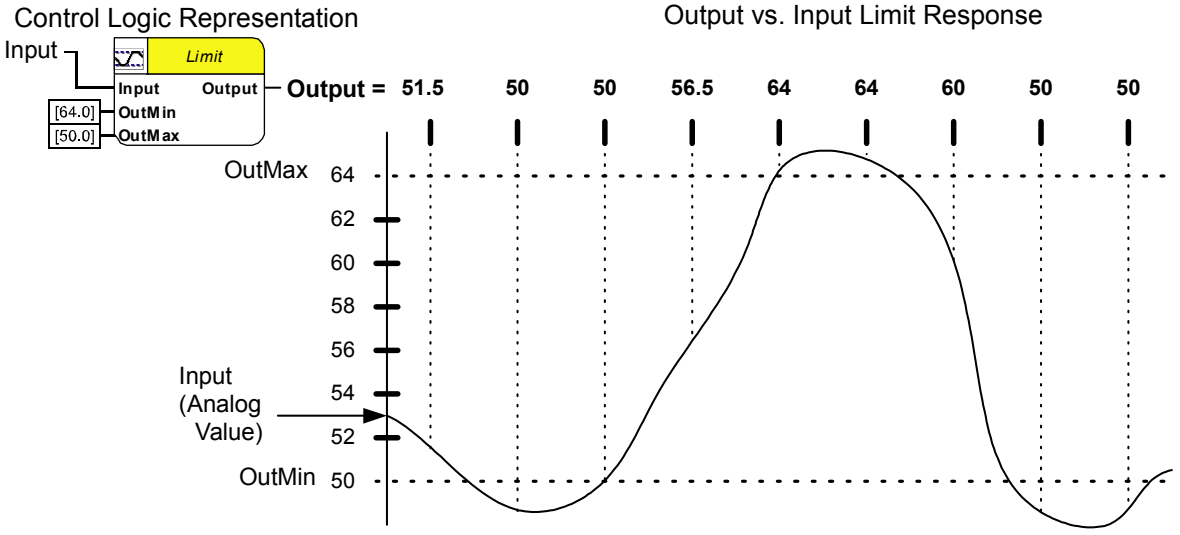
Table–5.174 Limit Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output reflecting the Input value, which is limited between the assigned Output Minimum and Output Maximum values. A value of not active (NA) indicates the Input is NA.	-163.83 to 16383

Applying the Object

The Limit object “clips” an input value (typically analog) at assigned maximum and minimum limits, but tracks the input when between these limits. Examples include limiting the output signal of a loop or limiting the range of setpoint values in an application.

Figure–5.80 shows an example of how the limit function is used in monitoring a varying analog value. The actual analog value at the Input varies between approximately 49 and 65 over a period of time. The output minimum low-limit is set for 50.0 and the output maximum high-limit is set for 64.0. The Output follows the input signal as long as the value remains between the two limit values assigned. This example shows how the Output does not exceed either high or low limit.



Figure–5.80 Example Input-to-Output Graph for a Limit Object.

Setting Output Maximum to a value less than the Output Minimum causes the Limit object to output the defined Output Maximum. Conversely, setting Output Minimum to a value which is greater than the Output Maximum value causes the Limit object to output the value defined by the Output Maximum.

Limit Thermostat

(Rev.3 or Later Firmware Required)

Object Usage: The Limit Thermostat object generates a digital ON / OFF output based upon high or low limit conditions at the Input. Trip points are established at the inputs Trip On and Trip Off. Depending on the values present at the Trip inputs, either a high-limit or low-limit thermostat function is provided, as follows:

- High-Limit Thermostat Function
  Trip On input value > Trip Off input value
- Low-Limit Thermostat Function
  Trip On input value < Trip Off input value

The Limit Thermostat object constantly monitors and compares the Input value to the Trip On and Trip Off values and sets the output accordingly. Upon controller reset, the output is initialized to the OFF state. The output is set to not active (NA) whenever any input (Input, Trip On, Trip Off) is NA.

Device Support: (See page 7)

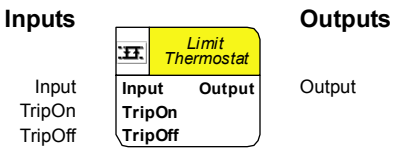
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F2, F3, H2, H3, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx
where xx = V2 or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.175 Limit Thermostat Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.

Table–5.175 Limit Thermostat Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.176 Limit Thermostat Object Input Properties

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Analog - The value monitored for limit thermostat control. This value is constantly compared to the Trip On and Trip Off values.	-163.83 to 16383	A not active (NA) at any input causes the Output to go to NA.
TripOn	Trip On	Class: Analog - This value defines the Input point to where the Output is set to the ON state.	-163.83 to 16383	
TripOff	TripOff	Class: Analog - This value defines the Input point to where the Output is set to the OFF state.	-163.83 to 16383	

Table–5.177 Limit Thermostat Object Output Properties

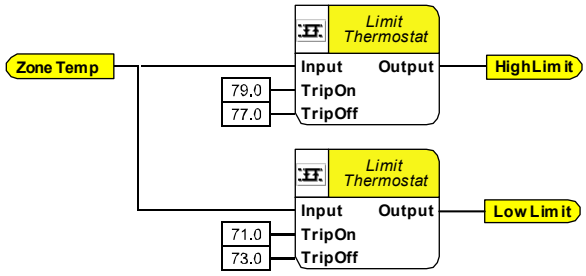
Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - Initialized to OFF at reset, then operates as follows: • High-Limit Operation (Trip On > Trip Off): Output set to ON whenever the Input exceeds the Trip On value. The output remains ON until the Input drops below the Trip Off value, whereby the output is returned to the OFF state. • Low-Limit Operation (Trip On <Trip Off): Output set to ON whenever the Input drops below the Trip On value. The output remains ON until the Input exceeds the Trip Off value, whereby the output is returned to the OFF state. In both cases, no change to the output occurs when the Input value is within the range defined by the Trip On and Trip Off values. The output is set to not active (NA) whenever any input has an NA.	OFF (0.0) ON (100.0)

Applying the Object

The Limit Thermostat object provides a simple limit-type thermostat function. Limits are defined by the relationship of the Trip On and Trip Off input values, which allows either a low-limit or high-limit configuration.

Example

In the example, below, two Limit Thermostat objects are used in an application designed to maintain a zone temperature setpoint of 75°, ± 4°.



Figure–5.81 Limit Thermostat Objects Used For High and Low Limit Control.

Anytime the system (zone temperature) goes outside this range, either the high or low limit output will be turned On. These outputs can be used for indication, control, or annunciation purposes. The “TripOff” values prevent oscillation at either limit point by providing hysteresis before a limit output returns Off.

Loop Sequenced

Object Usage: The Loop Sequenced object provides three sequenced loop outputs from a single object. Each loop monitors the same input, but has a separately available setpoint. The following loops are provided:

- Cooling (Loop1).
- Heating (Loop2).
- Economizer (Loop3).

The object ensures that its loop outputs remain sequenced in reference to each other, which prevents improper sequences (such as simultaneously active cooling and heating outputs). The cooling loop and heating loop each offer proportional (P), proportional plus integral (PI), or proportional plus integral plus derivative (PID) control action, based on the single sensed input and the individual control setpoint. The economizer loop provides proportional control with an adjustable minimum position and options for economizer overrides and automatic throttling range calculations. A ramp start feature is also available.

Device Support: (all firmware revisions)

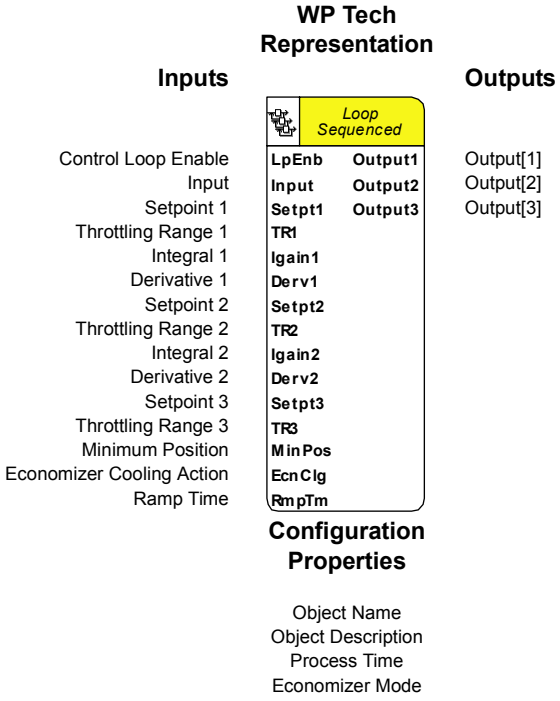
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 36 bytes
RAM: 56 bytes (standard controllers)
20 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Economizer Mode

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.178 Loop Sequenced Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.

Table–5.178 Loop Sequenced Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
EcnMod	Economizer Mode	Class: Analog - Defines the Economizer Mode (Loop 3) operation: either Controlled or Auto Range. • If set to Controlled, Loop 3 uses the Loop 3 inputs for Throttling Range (TR3) and Setpoint (Setpt3). • If set to Auto Range, Loop 3 uses a Throttling Range of Setpt1 - Setpt2 and a setpoint midway between.	0	0 - Controlled 1 - Auto Range	

Table–5.179 Loop Sequenced Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
LpEnb	Control Loop Enable	Class: Digital - An ON or not active (NA) allows the loop control algorithm to calculate and update the outputs. An OFF sets all outputs to 0.0%.	—	
Input	Input	Class: Analog - The sensed value of the media being controlled. This value is compared to the setpoint values and is used by the loop algorithm to calculate the output values.	-163.83 to 16383	A not active (NA) causes all outputs to be held to 0.0%.
Setpt1	Setpoint 1	Class: Analog - The control reference used by the cooling loop (Loop 1).	-163.83 to 16383	A not active (NA) causes Output[1] to be held at 0.0%. See the General Rules section for how setpoints interact.
TR1	Throttling Range 1	Class: Analog - The amount of input change needed for Loop 1 output to proportionally change from 0.0 to 100.0%. A value of 0.0, not active (NA), or a negative value causes Output[1] to be held at 0.0%.	0 to 16383	
Igain1	Integral 1	Class: Analog - The amount of Loop 1 integral gain expressed in repeats per minute. A value of 0.0, not active (NA), or a negative value disables the integral function.	0.00 to 10.00	
Derv1	Derivative 1	Class: Analog - The amount of Loop 1 derivative gain expressed in minutes. A value of 0.0, not active (NA), or a negative value disables the derivative function.	0.0 to 10.0	

Table–5.179 Loop Sequenced Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
Setpt2	Setpoint 2	Class: Analog - The control reference used by the heating loop (Loop 2).	-163.83 to 16383	A not active (NA) causes Output[2] to be held at 0.0%. See the General Rules section for how setpoints interact.
TR2	Throttling Range 2	Class: Analog - The amount of input change needed for Loop 2 output to proportionally change from 0.0 to 100.0%. A value of 0.0, not active (NA), or a negative value causes Output[2] to be held at 0.0%.	0 to 16383	
Igain2	Integral 2	Class: Analog - The amount of Loop 2 integral gain expressed in repeats per minute. A value of 0.0, not active (NA), or a negative value disables the integral function.	0.00 to 10.00	
Derv2	Derivative 2	Class: Analog - The amount of Loop 2 derivative gain expressed in minutes. A value of 0.0, not active (NA), or a negative value disables the derivative function.	0.0 to 10.0	
Setpt3	Setpoint 3	Class: Analog - The control reference used by the economizer loop (Loop 3) if EcnMode is set to Controlled. (Setpt3 is ignored if EcnMode is set to Auto Range.)	-163.83 to 16383	See the General Rules section for how setpoints interact and how not active (NA) is evaluated.
TR3	Throttling Range 3	Class: Analog - The amount of input change needed for Loop 3 output to proportionally change from 0.0 to 100.0% if EcnMode is set to Controlled. (TR3 is ignored if EcnMode is set to Auto Range.)	0 to 16383	
MinPos	Minimum Position	Class: Analog - Defines the minimum position or the minimum output allowed for the economizer (Loop 3) output when the Loop Sequence object is enabled.	0.0 to 100.0%	A not active (NA) sets minimum position to 0.0%.
EcnClg	Economizer Cooling Action	Class: Digital - Determines the economizer (Loop 3) action during cooling (Output 1 > 0.0%). • OFF or not active (NA) = Loop 3 output remains unaffected during cooling. • ON = Loop 3 output set to Minimum Position during cooling.	—	
RmpTm	Ramp Time	Class: Analog - Defines a loop ramp time (in minutes) in which all loop outputs are allowed to increase from 0.0% towards 100.0% from a loop enable (activation), following any of these events: • OFF-to-ON transition at Control Enable. • A controller reset. • Control Input change from not active (NA) to a valid value.	0.0 to 1,000 minutes	An NA or negative value is evaluated as 0.0 minutes (ramp function disabled).

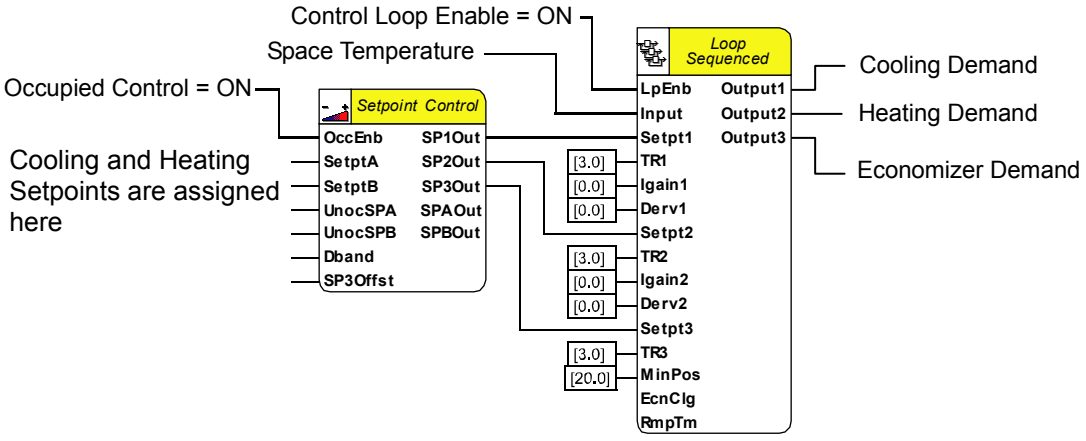
Table–5.180 Loop Sequenced Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output1	Output [1]	Class: Analog - The current calculated value of the cooling loop (Loop 1). The output is direct acting (increase in Input causes increase in Output). A not active (NA) at either the Input or Setpoint 1 holds the Output [1] value at 0.0%.	0.0 to 100.0%
Output2	Output [2]	Class: Analog - The current calculated value of the heating loop (Loop 2). The output is reverse acting (increase in Input causes decrease in Output). A not active (NA) at either the Input or Setpoint 2 holds the Output [2] value at 0.0%.	0.0 to 100.0%
Output3	Output [3]	Class: Analog - The current calculated value of the economizer loop (Loop 3). The output is direct acting (increase in Input causes increase in Output). A not active (NA) at the Input holds the Output [3] value at 0.0%.	0.0 to 100.0%

Applying the Object

This object is typically used for control strategies which include a cooling control loop, a heating control loop, and an economizer control loop. The cooling (Loop1) and heating (Loop2) control loops provide proportional, proportional plus integral, or proportional plus integral plus derivative control. The economizer (Loop3) control loop provides proportional style control plus additional features including adjustable minimum position, economizer override on call for cooling (Loop1), and automatic throttling range calculation. Loop outputs remain referenced to each other at all times, to protect against simultaneously active outputs.

All loops calculate their respective control outputs based upon the value at the object Input and their individually adjustable control setpoint. As shown in Figure–5.82 below, the Loop Sequenced object is typically paired with a Setpoint Control object (page 496) that provides separate cooling and heating setpoints during both occupied and unoccupied periods.



Figure–5.82 Loop Sequence Object Used With a Setpoint Control Object.

General Rules

To ensure proper loop sequencing, the setpoints for the three loops (Setpt1, Setpt2, and Setpt3) are evaluated in a priority fashion as follows:

- Setpt1 maintains priority over Setpt2 where Setpt2 is not allowed to exceed the Setpt1 value.
- Setpt1 maintains priority over Setpt3 where Setpt3 is not allowed to exceed the Setpt1 value.

Table–5.181 summarizes Loop Sequence object control conditions based on all setpoint combinations, including not active (NA) setpoint conditions.

Table–5.181 Loop Sequenced Object Setpoint Results.

Setpoint 1	Setpoint 2	Setpoint 3	Control Conditions and Results
Valid	Valid	Valid	If Setpoint 2 > Setpoint 1, then Setpoint 2 = Setpoint 1. If Setpoint 3 > Setpoint 1, then Setpoint 3 = Setpoint 1.
Valid	Valid	NA	If Setpoint 2 > Setpoint 1, then Setpoint 2 = Setpoint 1. Output[3] is held at 0.0%.
Valid	NA	Valid	If Setpoint 3 > Setpoint 1, then Setpoint 3 = Setpoint 1. Output[2] is held at 0.0%.
Valid	NA	NA	Output[2] and Output[3] are held at 0.0%.
NA	Valid	Valid	Output[1] is held at 0.0%.
NA	Valid	NA	Output[1] and Output[3] are held at 0.0%.
NA	NA	Valid	Output[1] and Output[2] are held at 0.0%.
NA	NA	NA	All outputs are held at 0.0%.

Note also that a loop Throttling Range of not active (NA), zero or a negative value causes that loop's output to be held at 0.0%. For example if the Loop1 Throttling Range (TR1) is NA, then Output[1] is held at 0.0%.

Ramp Start

Sequenced startup ramping (soft start) is available and is applied to all control loops when the object is initialized from a controller reset or object enable. A Control Loop Enable of OFF or an Input of not active (NA) causes all three loop outputs to default to 0.0%.

The ramp function is initiated for the following conditions:

- The controller is reset.
- Enable of the Control Loop Enable input from OFF to ON.
- The Input value changes from not active (NA) to a valid value.

The ramp function ramps the outputs in a direction determined by the present heat / cool demand.

- A demand for heat causes the Loop2 output to ramp from zero (0.0%) towards full demand (100.0%) until the demand is satisfied.
- A demand for economizer / cooling causes the Loop3 economizer output to ramp from Minimum Position towards 100.0% followed by the Loop1 cooling output to ramp from 0.0% towards 100.0% until the economizer / cooling demands are satisfied.

The rate of ramp for all loops is defined by the Ramp Time property.

Loop Differences

The cooling loop (Loop1) and heating loop (Loop2) each have inputs for integral gain (Igain1 and Igain2) and derivative term (Derv1 and Derv2). Both of these loops operate in a similar (but opposite - Direct/ Reverse) fashion, with their output at 0.0% with the Input at their respective setpoint.

The economizer loop (Loop3) provides proportional only control, but has additional properties that affect the loop's throttling range and setpoint operation. These Loop3 properties are explained in the "Economizer (Loop3)" section ahead.

All three loops (Loop1, Loop2, and Loop3) follow the loop general rules given in Table–5.181. The only exception is for the economizer loop (Loop3), and only if the Loop Sequence object is set with an Economizer Mode of Auto Range. In this case, the object inputs TR3 and Setpt3 are ignored. See the "Economizer (Loop3)" section (page 321) for an explanation of these special "Economizer Rules".

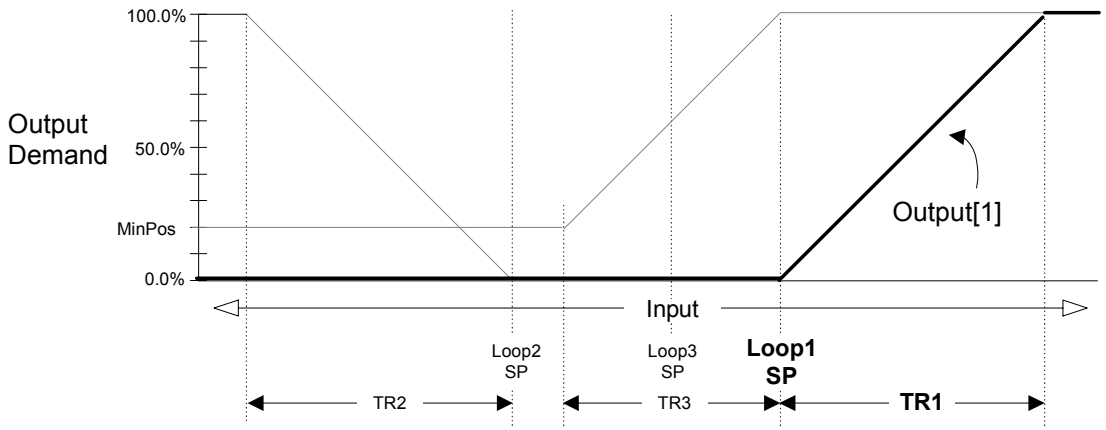
Cooling (Loop1)

The cooling loop (Loop1) provides proportional, proportional plus integral, or proportional plus integral plus derivative control for cooling applications. Properties specific to Loop1 operation include Setpt1, TR1, Igain1, and Derv1 with the output for Loop1 provided through Output[1].

Proportional Control

With proportional control, a control signal, based on the difference between Input and Setpt1, is produced. The difference, such as that between an actual temperature and setpoint, is the "error." Loop1 creates an output signal directly proportional to the error's magnitude. The relationship between the error and the output is controlled by the assigned Throttling Range (TR1). The Throttling Range value is the amount of change required at the Input to cause the output to go from 0.0% to 100.0%.

For proportional control, Output[1] is set to 0.0% when the Input is equal to the Setpt1 value. Control action is direct acting, which means that the objects output signal increases (advances towards 100.0%) as the input to the Loop1 algorithm increases above Setpt1. The calculated Output[1] percent value is the difference between the Input value and the Setpt1 value, divided by the Throttling Range (TR1) times 100, as shown in Figure–5.83.



Figure–5.83 Cooling (Loop1) Output in Relation to Other Loops.

Integral Control

A characteristic of proportional only control is that it exhibits an offset or droop (error) condition as the output moves through its throttling range. Because of this, proportional only control is unable to maintain an exact setpoint. The use of integral action is designed to eliminate offset. An integrating term (Igain1) is used to observe how long the error condition has existed, summing the error over time. The summation value becomes the basis for an additional control signal, which is added to the signal produced by the proportional term. The control loop continues to produce a control action over time, allowing the elimination of offset.

Proportional-integral (PI) control can:

- Respond to the presence of error in the control loop.
- Relate to the magnitude of the control signal to that of the error.
- Respond to offset over time to achieve zero error at Setpoint.

When using PI control, the Integral and Throttling Range values must be carefully sized to minimize overshoot, often present in modulating control loops. Overshoot refers to a control loop's tendency to overcompensate for an error condition, causing a new error in the opposite direction. In some instances, loop overshoot can repeat itself in an oscillating fashion. See the "Loop Control Guidelines" section for details on establishing values for the Throttling Range and Integral inputs.

The object input Derivative (Derv) is not used in PI control and should be assigned to 0.0, or may be left unconnected (not active, NA).

Derivative Control

In response to overshoot, derivative action provides an anticipatory function that exerts a "braking" action on the control loop. The derivative term (Derv1) is based on the error's rate of change. The derivative function observes how fast the actual condition approaches the desired condition, producing a control action, based on this rate of change.

This additional Derivative action anticipates the convergence of actual and desired conditions, in effect, counteracting the control signal produced by the Proportional and Integral terms. Properly applied, the result is a significant reduction in overshoot. However, the Proportional, Integral, and Derivative actions can be tricky to apply, with the Derivative action able to produce unexpected results. For this reason, most HVAC loop applications use PI control.

Ramp Function in Cooling Demand

Upon completion of the economizer output ramp, the ramp function causes the cooling loop output value to ramp at a rate specified by the ramp time from 0.0% towards 100.0% after the loop is activated or enabled into PID control. The ramp function will terminate when:

- The calculated Output[1] is equal to 0.0% prior to ramp initialization.
- The actual ramped Output[1] value equals the calculated Output[1].
- The actual ramped Output[1] value reaches the maximum output value of 100.0%.

Heating (Loop2)

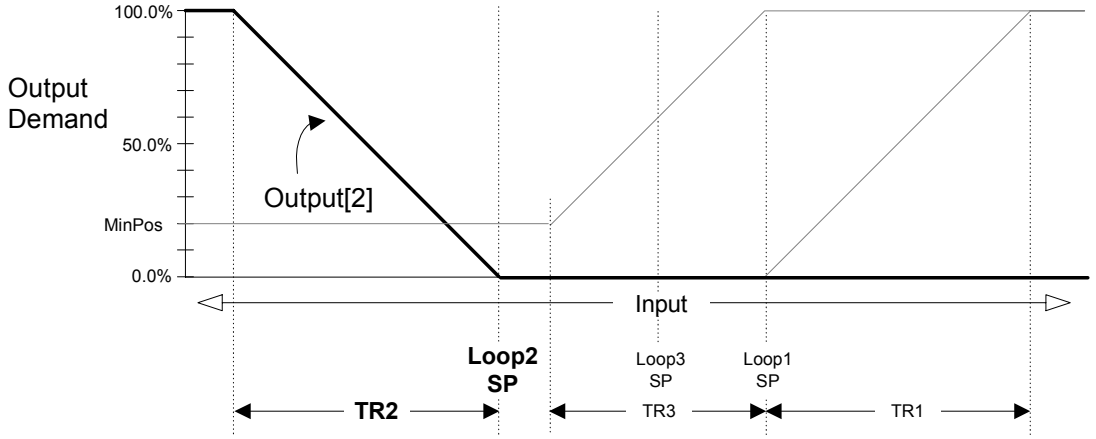
The heating loop (Loop2) provides proportional, proportional plus integral, or proportional plus integral plus derivative control for heating applications. Properties specific to Loop2 operation include Setpt2, TR2, Igain2, and Derv2 with the output for Loop2 provided through Output[2].

Proportional control

With proportional control, a control signal, based on the difference between Input and Setpt2, is produced. The difference, such as that between an actual temperature and setpoint, is the “error.” Loop2 creates an output signal directly proportional to the error’s magnitude.

The relationship between the error and the output is controlled by the assigned Throttling Range (TR2). The Throttling Range value is the amount of change required at the Input to cause the output to go from 0.0% to 100.0%.

For proportional control, Output[2] is set to 0.0% when the Input is equal to the Setpt2 value. Control action is reverse-acting, which means that the objects output signal increases (advances towards 100.0%) as the input to the Loop2 algorithm decreases below Setpt2. The calculated Output[2] percent value is the difference between the Setpt2 value and the Input value, divided by the Throttling Range (TR2) times 100, as shown in Figure–5.84.



Figure–5.84 Heating (Loop2) Output in Relation to Other Loops.

Integral control

A characteristic of proportional only control is that it exhibits an offset or droop (error) condition as the output moves through its throttling range. Because of this, proportional only control is unable to maintain an exact setpoint. The use of integral action is designed to eliminate offset droop. An integrating term (Igain2) is used to observe how long the error condition has existed, summing the error over time. The summation value becomes the basis for an additional control signal, which is added to the signal produced by the proportional term. The control loop continues to produce a control action over time, allowing the elimination of offset.

Proportional-integral (PI) control can:

- Respond to the presence of error in the control loop.
- Relate to the magnitude of the control signal to that of the error.
- Respond to offset over time to achieve zero error at Setpoint.

When using PI control, the Integral and Throttling Range values must be carefully sized to minimize overshoot, often present in modulating control loops. Overshoot refers to a control loop's tendency to overcompensate for an error condition, causing a new error in the opposite direction. In some instances, loop overshoot can repeat itself in an oscillating fashion. See the "Loop Control Guidelines" section for details on establishing values for the Throttling Range and Integral inputs.

The object input Derivative (Derv2) is not used in PI control and should be assigned to 0.0, or may be left unconnected (not active, NA).

Derivative Control

In response to overshoot, derivative action provides an anticipatory function that exerts a "braking" action on the control loop. The derivative term (Derv2) is based on the error's rate of change. The derivative function observes how fast the actual condition approaches the desired condition, producing a control action, based on this rate of change.

The Derivative action anticipates the convergence of actual and desired conditions, in effect, counteracting the control signal produced by the Proportional and Integral terms. Properly applied, the result is a significant reduction in overshoot. However, Proportional, Integral, and Derivative actions can be tricky to apply, and Derivative action is able to produce unexpected results. For this reason, most HVAC-loops use PI control.

Ramp Function in Heating Demand

The ramp function causes the heating loop output value to ramp at a rate specified by the ramp time from 0.0% towards 100.0% after the loop is activated or enabled. The ramp function will terminate when:

- The calculated Output[2] is equal to 0.0% prior to ramp initialization.
- The actual ramped Output[2] value equals the calculated Output[2].
- The actual ramped Output[2] value reaches the maximum output value of 100.0%.

Economizer (Loop3)

The economizer (Loop3) provides control for economizer applications. Properties specific to Loop3 operation include Setpt3, TR3, MinPos, and EcnClg, with the output for Loop3 provided through Output[3]. The configuration property EcnMode (Economizer Mode) determines the economizer sequence followed: Controlled or Auto Range.

Controlled

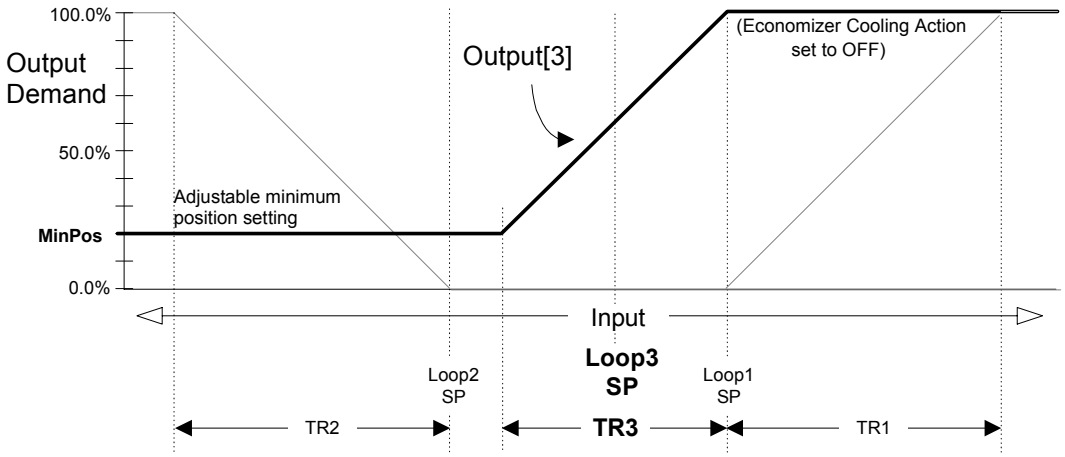
An Economizer Mode selection of Controlled causes the economizer to operate using a standard proportional style control with a control signal based on the difference between Input and Setpt3. The difference, such as that between an actual temperature and setpoint, is the “error.” Loop3 creates an output signal directly proportional to the error’s magnitude. The relationship between the error and the output is controlled by the assigned Throttling Range (TR3). The Throttling Range value is the amount of change required at the Input to cause the output to operate from Minimum Position to 100.0%.

For proportional control, Output[3] is set to a midpoint position which is directly between Minimum Position and 100.0% when the Input is equal to the Setpt3 value. Control action is direct-acting, meaning that the object’s output signal increases (advances towards 100.0%) as the input to the Loop3 algorithm increases. Minimum Position is the minimum output allowed for the economizer output when the Loop Sequenced object is enabled.

Economizer Cooling Action Input

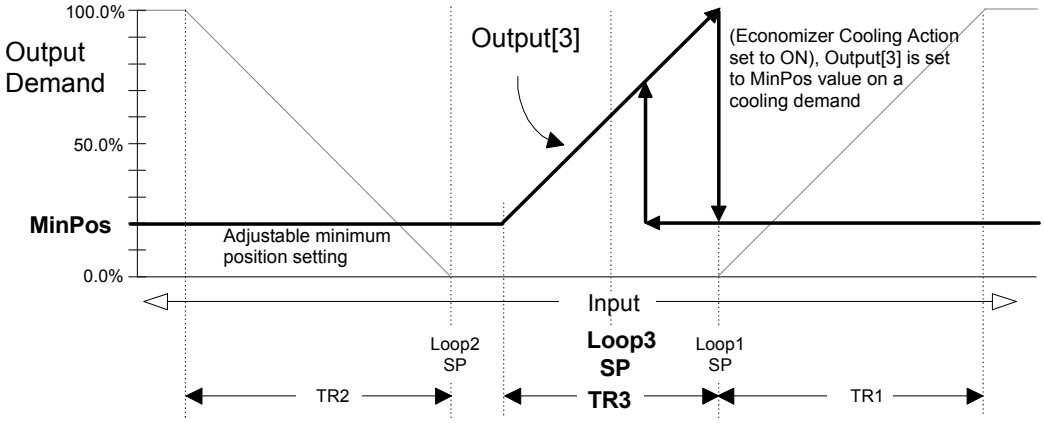
In the Controlled mode, the Economizer Cooling Action input defines the action on a call for active cooling or Output[1] is greater than 0.0%.

Set to OFF: Economizer Cooling Action set to OFF causes the economizer output to remain unaffected during operation within the active cooling range, as shown below in Figure–5.85.



Figure–5.85 Loop3 (Economizer) Output with EncMode = Controlled and EcnClg = OFF.

Set to ON: Economizer Cooling Action set to ON causes the economizer output to be set to the Minimum Position value when the cooling output is within the active range or greater than 0.0%, as shown in Figure–5.86 below. The economizer output will then remain overridden at Minimum Position until the cooling Output[1] returns to 0.0% and the economizer demand is less than or equal to 2/3 of the economizer TR3.



Figure–5.86 Loop3 (Economizer) Output with EncMode = Controlled and EcnClg = ON.

The output value is calculated using the following equations:

Result1 = ([(Input - Setpt3) + (0.5 * TR3)] ÷ TR3) x 100

Note: Result1 is limited between 0% and 100%.

Output[3] = [(100 - Min Position) x (Result1 ÷ 100)] + Min Position

Auto Range

An Economizer Mode selection of Auto Range causes the economizer to operate using a standard proportional style control with a control signal based upon the cooling (Loop1) and heating (Loop2) setpoints. Properties specific to Loop3 operation including Setpt3 and TR3 are ignored. The MinPos and EcnClg properties remain available for use.

The difference between the actual temperature and setpoint is the “error.” Loop3 creates an output signal directly proportional to the error’s magnitude. The relationship between the error and the output is controlled by the calculated setpoint (SPx) and Throttling Range (TRx). SPx is calculated to be the setpoint midway between the cooling Loop1 (Setpt1) and heating Loop2 (Setpt2) setpoint values.

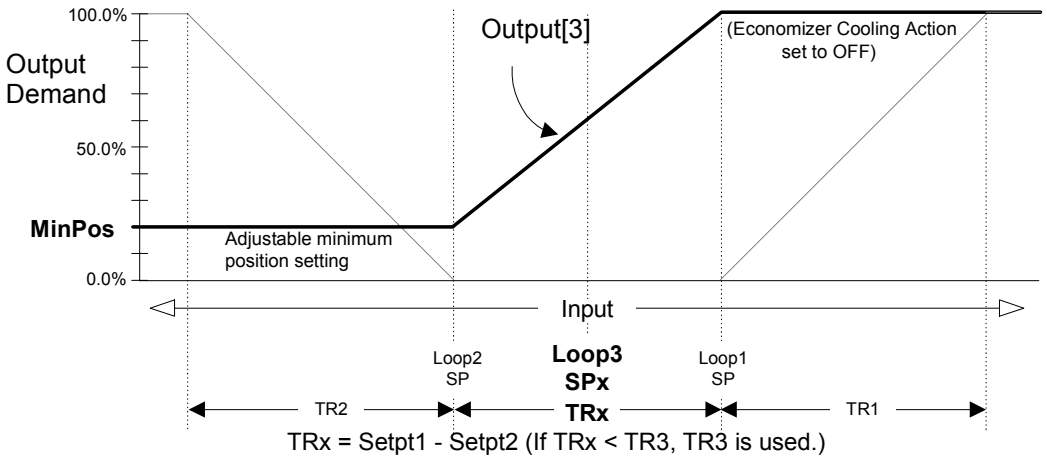
The Throttling Range (TRx) value is the calculated amount of change required at the Input to cause the output to operate from Minimum Position to 100.0%. TRx is calculated to be the difference between the cooling Loop1 (Setpt1) and heating Loop2 (Setpt2) setpoint values.

Control action is direct, which means that the objects output signal increases (advances towards 100.0%) as the input to the Loop3 algorithm increases. Minimum Position is the minimum output allowed for the economizer output when the Loop Sequenced object is enabled.

Economizer Cooling Action Input

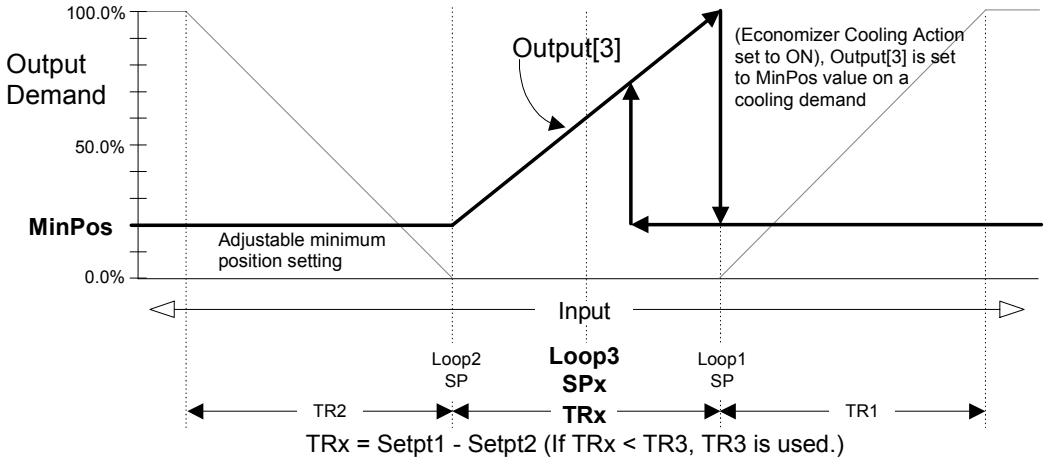
In the Auto Range mode, the Economizer Cooling Action Input defines the action on a call for active cooling or Output[1] is greater than 0.0%.

Set to OFF: Economizer Cooling Action set to OFF causes the economizer output to remain unaffected during operation within the active cooling range, as shown in Figure–5.87 below.



Figure–5.87 Loop3 (Economizer) Output with EncMode = Auto Range and EcnClg = OFF.

Set to ON: Economizer Cooling Action set to ON causes the economizer output to be set to the Minimum Position value when the cooling output is within the active range or greater than 0.0%, as shown in Figure–5.88 below. The economizer output will then remain overridden at Minimum Position until the cooling Output[1] returns to 0.0% and the economizer demand is less than or equal to 2/3 of the economizer TRx.



Figure–5.88 Loop3 (Economizer) Output with EncMode = Auto Range and EcnClg = ON.

The Loop3 output value is calculated using the following equations:

SPx = [(Setpt1 - Setpt2) ÷ 2] + Setpt2
TRx = Setpt1 - Setpt2

Note: Setpt2 is never allowed to exceed Setpt1.
Also, if TRx < TR1, then TRx = TR1.

Result1 = ([(Input - SPx) + (0.5 x TRx)] ÷ TRx) x 100

Note: Result1 is limited between 0% and 100%.

Output[3] = [(100 - Min Position) x (Result1 ÷ 100)] + Min Position

Ramp Function Economizer Demand

The ramp function causes the economizer loop output value to ramp at a rate specified by the ramp time from 0.0% towards 100.0% after the Sequence Loop object is activated or enabled.

The ramp function terminates when:

- The calculated Output[3] is equal to 0.0% prior to ramp initialization.
- The actual ramped Output[3] value equals the calculated Output[3].
- The actual ramped Output[3] value reaches the maximum output value of 100.0%.

Economizer Rules

If the Economizer Mode is set to Controlled, the Loop Sequence object observes all “General Rules” for setpoint priority and control conditions, as previously shown Table–5.181.

If the Economizer Mode is set to Auto Range, the Setpt3 and TR3 properties are not used. The Loop3 Throttling Range (TRx) and Loop3 Setpoint (SPx) are based on values assigned to Setpt1, Setpt2, and TR1, and are calculated using the following:

TRx = Setpt1 - Setpt2
(If TRx < TR1, then TRx = TR1).

Output [3] ranges from minimum position to 100% over TRx with Loop3 midpoint at SPx.

If the Loop1 Throttling Range (TR1) is not active (NA), zero, or a negative value, Output[3] is held at 0.0%. The following table shows control conditions based on all setpoint combinations for a Loop Sequence object set to an Economizer Mode of Auto Range:

Table–5.182 Setpoint Results if Economizer Mode = Auto Range.

Setpoint 1	Setpoint 2	Control Conditions and Results
Valid	Valid	If Setpt2 > Setpt1, then Setpt2 = Setpt1
Valid	NA	TRx = TR1 Output[2] is held at 0.0%
NA	Valid	Output[1] and Output[3] are held at 0.0%.
NA	NA	All outputs are held at 0.0%.

General Guidelines for Setting Up Loop Control

1. Using only proportional control, adjust the Throttling Range (TR) value until the loop control is stable with no oscillation. Do not hesitate to increase the TR if necessary because some loops, such as mixed air, may require a throttling range of 25°F or more to achieve stability. If stability can not be achieved, the mechanical system installation and design should be reviewed. If stability can not be achieved with proportional control, integral and/or derivative will cause further instability.
2. Before stability is achieved, in step #1, measure the period of oscillation. This is the amount of time, in minutes, from one peak to the next.
3. Once stability is achieved by using proportional only, proceed to increase the Throttling Range value by 20% to 50% in preparation for adding integral.
4. Use the following formula to calculate the integral value to be used:

   i = 1 ÷ [(Loop Period in Minutes) x 2]

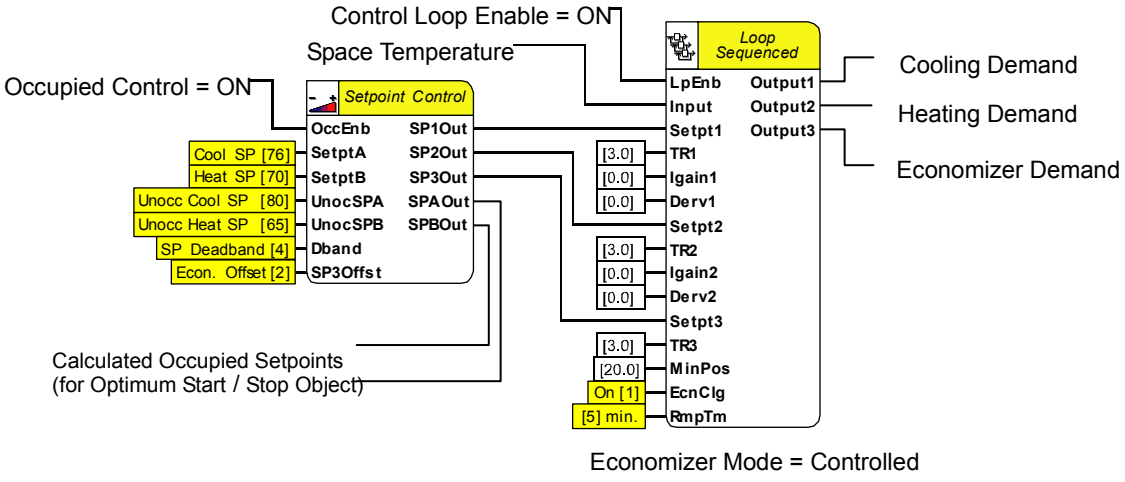
   This formula yields a good starting point for integral action.
5. Monitor loop control to evaluate response. If the response is slow with integral action, increase the “i” values slightly. It may be necessary, to upset the loop to get a good test of the loop response. This could be done by changing the setpoint to simulate a sudden change in the load and then observe the time required to reach the new setpoint. In general, it is recommended that Integral not exceed 1.0. Typically, values between 0.05 and 0.5 are usually effective.
6. In most cases, the control loops used in the HVAC industry do not require derivative action. It is recommended not to use the derivative action since an improper value is worse than none at all. If derivative is required, use the following formula to determine the proper value:

   d = Loop Period in Minutes ÷ 8

   This formula yields a good starting point for derivative action.

Example Application

The following example illustrates a method for combining the Setpoint Control object with the Loop Sequenced object to provide sequenced cooling, heating, and economizer control. The Setpoint Control object is configured for dual setpoint control which causes the object to utilize a separate cooling and heating occupied setpoint to generate occupied cooling, heating, and economizer setpoints.



Figure–5.89 Example Loop Sequence Object with Setpoints Supplied by a Setpoint Control Object.

In this example, the following constant values are assigned to the inputs of the Setpoint Control object:

Cool SP = 76.0

Heat SP = 70.0

Unoccupied Cool SP = 80.0

Unoccupied Heat SP = 65.0

Deadband = 4.0

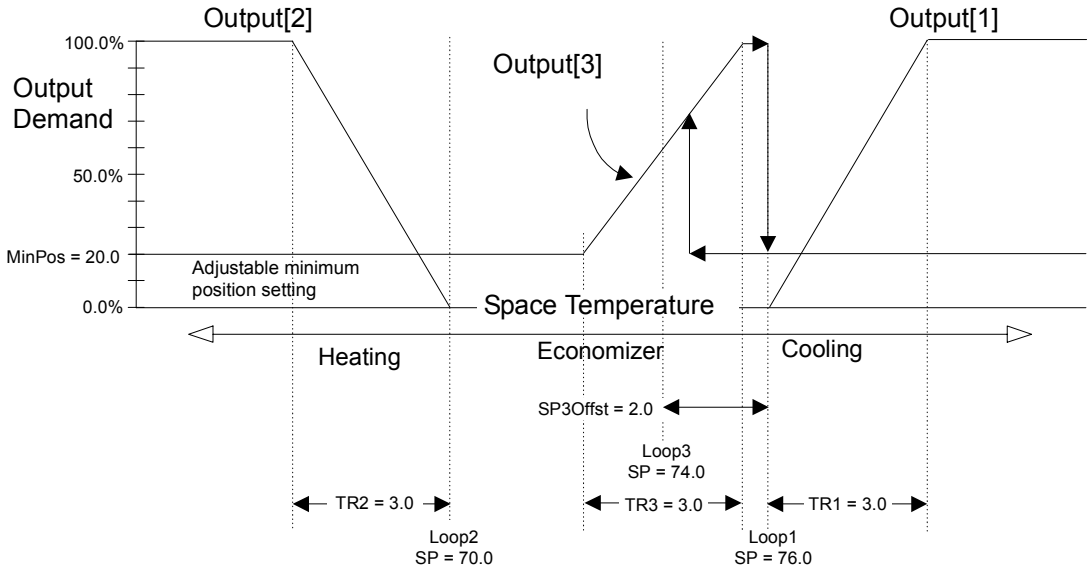
SP3Offst = 2.0

Based upon the input values supplied to the Setpoint Control object, the following setpoint values are generated:

Setpoint Control Outputs	OccEnb = ON (Occupied)	OccEnb = OFF (Unccupied)
SP1Out (Cool Setpoint)	76.0	80.0
SP2Out (Heat Setpoint)	70.0	65.0
SP3Out (Economizer Setpoint)	74.0	78.0
SPAOut (Occupied Cool Setpoint)	76.0	76.0
SPBOut (Occupied Heat Setpoint)	70.0	70.0

Deadband prevents heating / cooling setpoint crossover by maintaining a 4.0 deadband. The SP3Offst and Deadband properties have been setup to insure a complete economizer operating range between the heating and cooling cycles.

The following illustration shows the Loop Sequenced output demand characteristics for this example's cooling, heating, and economizer operation during occupied mode.



Figure–5.90 Loop Sequenced Control (with EcnMod = Controlled and EcnClg = ON).

Loop Single

Object Usage: The Loop Single object provides either proportional (P), proportional plus integral (PI), or proportional plus integral plus derivative (PID) control action of a controlled output, based on a sensed input and a control setpoint. The loop action may be switched between direct acting and reverse acting. Also provided is an adjustable output reference and a ramp start feature.

Device Support: (all firmware revisions)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

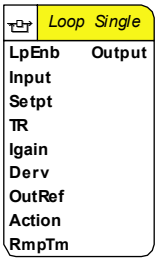
Memory Requirements: (per object)

EEPROM: 22 bytes
RAM: 38 bytes (standard controllers)
16 bytes (MN 800)

WP Tech Representation

Inputs

Control Loop Enable
Input
Setpoint
Throttling Range
Integral
Derivative
Output Reference
Action
Ramp Time



Outputs

Output

Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.183 Loop Single Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.184 Loop Single Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
LpEnb	Control Loop Enable	Class: Digital - An ON or not active (NA) allows the loop control algorithm to calculate and update the Output. An OFF sets the Output to 0.0%.	—	

Table–5.184 Loop Single Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Analog - The sensed value of the media being controlled. This value is compared to the Setpoint value and is used by the loop algorithm to calculate the Output value.	-163.83 to 16383	A not active (NA) causes the Output to be held to 0.0% for direct acting or 100% for reverse acting.
Setpt	Setpoint	Class: Analog - The control reference used by the loop algorithm.	-163.83 to 16383	A not active (NA) causes the Output to be held to 0.0%
TR	Throttling Range	Class: Analog - The amount of input change needed to cause the Output value of the object to change from 0.0 to 100.0%. A value of 0.0, not active (NA), or a negative value causes the Output to be held at 0.0%.	0 to 16383	
Igain	Integral	Class: Analog - The amount of integral gain expressed in repeats per minute. A value of 0.0, not active (NA), or a negative value disables the integral function.	0.00 to 10.00	
Derv	Derivative	Class: Analog - The amount of derivative gain expressed in minutes. A value of 0.0, not active (NA), or a negative value disables the derivative function.	0.00 to 10.00	
OutRef	Output Reference	Class: Analog - This input defines the output value when the input is equal to the value at the setpoint for proportional segment of the loop algorithm. A not active (NA) or a negative value is evaluated as 0.0%.	0.0 to 100.0%	
Action	Action	Class: Digital - Determines if the loop response is direct acting or reverse acting. • OFF or not active (NA) = direct acting. • ON = reverse acting.	ON, OFF	
RmpTm	Ramp Time	Class: Analog - Defines a loop ramp time (in minutes) in which the loop Output is allowed to increase from 0.0% towards 100.0% from a loop enable (activation), following any of these events: • OFF-to-ON transition at Control Enable. • A controller reset. • Control Input change from NA to a valid value.	0.0 to 1,000 minutes	An NA or negative value is evaluated as 0.0 minutes (ramp function disabled).

Table–5.185 Loop Single Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The current calculated value of the loop algorithm. A not active (NA) at either the Input or the Setpoint holds the Output value at 0.0%.	0.0 to 100.0%

Applying the Object

Depending on how the object inputs are used, the Loop Single object provides proportional (P), proportional plus integral (PI), or proportional plus integral plus derivative (PID) loop control action. Typical HVAC loops use PI control for reasons explained ahead. Each of the three loop control methods using the Loop Single object are summarized below.

Proportional only (P) Control

Proportional control is used for conventional closed loop control systems. With proportional control, a control signal, based on the difference between an actual condition (Input) and a desired condition (Setpoint), is produced. The difference, such as that between an actual temperature and setpoint, is the “error.” The Loop Single object creates an Output value directly proportional to the error’s magnitude. The relationship between the error and the output is controlled by the assigned Throttling Range (TR). The Throttling Range value is the amount of change required at the Input to cause the Output to go from 0.0% to 100.0%.

Control action (Action) can be switched between direct or reverse acting. Direct acting (the default) means that the object’s Output signal increases (towards 100.0%) as the Input to the Loop Single object increases. Reverse acting means that the object’s Output signal decreases (towards 0.0%) as the Input to the Loop Single object increases.

The Output Reference is the Output value at which the Input is equal to the Setpoint in proportional control (typically 50.0%). For proportional only control, the actual Output value is equal to the following:

Direct acting: Output% = [((input-setpt)÷TR) X 100] + Output Reference%

Reverse acting: Output% = [((setpt - input)÷TR) X 100] + Output Reference%

Object inputs Integral (Igain) and Derivative (Derv) are not used in Proportional only control and should be assigned to 0.0, or may be left unconnected (not active, NA).

Proportional plus Integral (PI) Control

A characteristic of Proportional only (P) control is that it exhibits an offset or droop (error) condition as the output moves through its throttling range. Because of this, Proportional only control is unable to maintain a desired condition, except at the exact Output Reference condition (typically 50%). The use of integral action is designed to eliminate offset droop. An integrating term can be used to observe how long the error condition has existed, summing the error over time. The summation value becomes the basis for an additional control signal, which is added to the signal produced by the proportional term. The control loop continues to produce a control action over time, allowing the elimination of offset.

A proportional-integral (PI) control can:

- Respond to the presence of error in the control loop.
- Relate to the magnitude of the control signal to that of the error.
- Respond to offset over time to achieve zero error at Setpoint.

When using PI control, the Integral and Throttling Range values must be carefully sized to minimize overshoot, often present in modulating control loops. Overshoot refers to a control loop’s tendency to overcompensate for an error condition, causing a new error in the opposite direction. In some instances, loop overshoot can repeat itself in an oscillating fashion. See the “Loop Control Guidelines” section for details on establishing values for the Throttling Range and Integral inputs.

The object input Derivative (Derv) is not used in PI control and should be assigned to 0.0, or may be left unconnected (not active, NA).

Proportional plus Integral plus Derivative (PID) Control

In response to overshoot, Derivative action provides an anticipatory function that exerts a “braking” action on the control loop. The derivative term is based on the error’s rate of change. The derivative function observes how fast the actual condition approaches the desired condition, producing a control action, based on this rate of change.

This additional Derivative action anticipates the convergence of actual and desired conditions, in effect, counteracting the control signal produced by the Proportional and Integral terms. Properly applied, the result is a significant reduction in overshoot. However, the Proportional, Integral, and Derivative actions can be tricky to apply, with the Derivative action able to produce unexpected results. For this reason, most HVAC loop applications use PI control.

Action

If the Action input is not active (NA) or OFF, loop response is direct acting; if the Action input is ON, loop response is reverse acting.

- Direct acting is where the Output value increases as the Input value increases, as in a typical cooling temperature loop.
- Reverse acting is where the Output value decreases as the Input value increases, as in a typical heating temperature loop.

Ramp Start Function

The ramp start function causes the loop Output value to ramp at a rate specified by the Ramp Time from 0.0% towards 100.0% after the Loop Single object is activated or enabled into PID control.

The ramp start function is initiated for the following conditions:

- The controller is reset.
- Enable of the Control Loop Enable input from OFF to ON.
- The Input value changes from not active (NA) to a valid value.

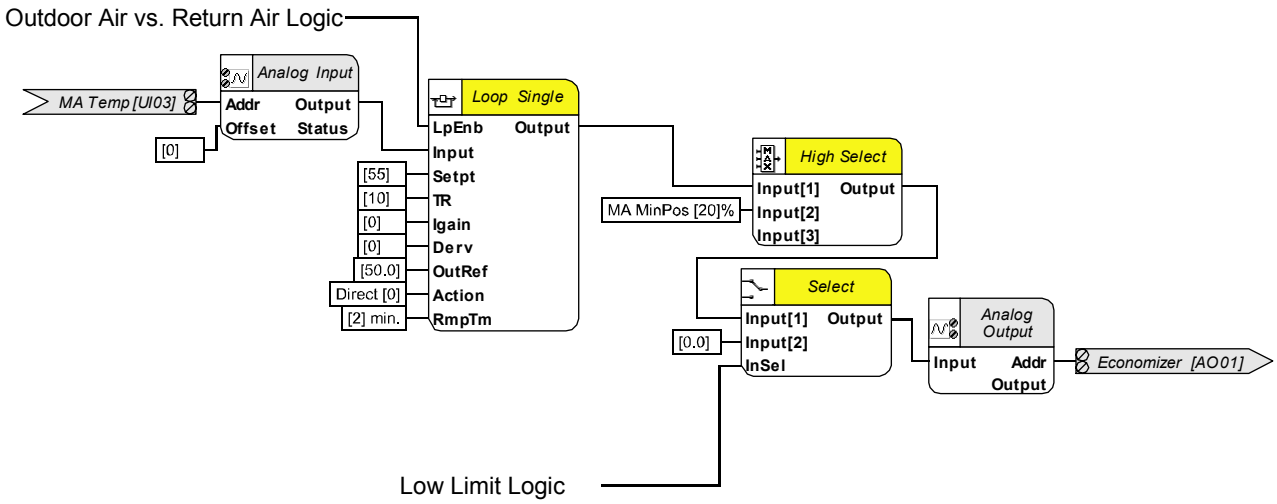
The ramp start function terminates when:

- The calculated output target is equal to 0.0% prior to initialization of the ramp start function.
- The actual ramped Output value equals the calculated output target.
- The actual ramped Output value reaches the maximum Output value of 100.0%.

Example

The Loop Single object shown in Figure–5.91 is used for mixed-air temperature control of a proportional economizer in a roof top unit. The object is enabled by a digital ON result of some logic based on outside air temperature (not shown). The value of the mixed air temperature sensor is compared against the desired loop Setpoint, which in this case is a constant 55°F. Loop response is proportional only, as both the Integral and Derivative inputs are assigned to 0 (zero). The Throttling Range is 10°F and the Output Reference is at 50%. A two minute Ramp Time is also assigned.

The Output value of the Loop Single object ultimately feeds an Analog Output object that modulates the proportional economizer. (In this example, first a High Select object establishes a minimum 20% position, and a Select object processes any binary low-limit temperature signal).



Figure–5.91 Example Loop Single Object Used for Mixed Air Proportional Economizer Control.

General Guidelines for Setting Up Loop Control

1. Using only proportional control, adjust the Throttling Range (TR) value until the loop control is stable with no oscillation. Do not hesitate to increase the TR if necessary because some loops, such as mixed air, may require a throttling range of 25°F or more to achieve stability. If stability can not be achieved, the mechanical system installation and design should be reviewed. If stability can not be achieved with proportional control, integral and/or derivative will cause further instability.
2. Before stability is achieved, in step #1, measure the period of oscillation. This is the amount of time, in minutes, from one peak to the next.
3. Once stability is achieved by using proportional only, proceed to increase the Throttling Range value by 20% to 50% in preparation for adding integral.
4. Use the following formula to calculate the integral value to be used:

   i = 1 ÷ [(Loop Period in Minutes) x 2]

   This formula yields a good starting point for integral action.
5. Monitor loop control to evaluate response. If the response is slow with integral action, increase the “i” values slightly. It may be necessary, to upset the loop to get a good test of the loop response. This could be done by changing the setpoint to simulate a sudden change in the load and then observe the time required to reach the new setpoint.

   In general, it is recommended that Integral not exceed 1.0. Typically, values between 0.05 and 0.5 are usually effective.
6. In most cases, the control loops used in the HVAC industry will not require derivative action. It is recommended not to use the derivative action since an improper value is worse than none at all. If derivative is required, the following formula can be used to determine the proper value.

   d = Loop Period in Minutes ÷ 8

This formula yields a good starting point for derivative action.

Low Select

Object Usage: The Low Select object acts as a low-signal selector that outputs the lowest value present on the three inputs. The Low Select object is typically used with analog values.

Output = Lowest Input[1], [2], or [3]

The Low Select object is the functional opposite of the High Select object.

Device Support: (all firmware revisions)

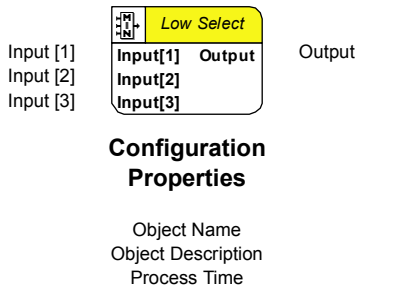
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controllers)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.186 Low Select Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.187 Low Select Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The first input evaluated for the lowest value. A not active (NA) is not evaluated.	-163.83 to 16383	If not active (NA) is present at all Inputs, the output is set to NA.
Input[2]	Input [2]	Class: Analog - The second input evaluated for the lowest value. A not active (NA) is not evaluated.	-163.83 to 16383	
Input[3]	Input [3]	Class: Analog - The third input evaluated for the lowest value. A not active (NA) is not evaluated.	-163.83 to 16383	

Table–5.188 Low Select Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output reflects the lowest input value. A value of not active (NA) indicates that all inputs are NA.	-163.83 to 16383

Applying the Object

The Low Select object outputs the lowest valid value present on the inputs. Not active (NA) inputs are ignored unless all inputs are NA, in which case the Output is NA. Table–5.189 below shows all possible input to output combinations, including NA input conditions.

Table–5.189 Truth Table for Low Select Object.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	Lowest of (AV1, AV2, AV3)
AV1	AV2	NA	Lowest of (AV1, AV2)
AV1	NA	AV3	Lowest of (AV1, AV3)
NA	AV2	AV3	Lowest of (AV2, AV3)
AV1	NA	NA	AV1
NA	AV2	NA	AV2
NA	NA	AV3	AV3
NA	NA	NA	NA

Inputs are typically analog values provided by another objects output or from an assigned constant. However, Inputs can also process numerical representations of digital values (0 for OFF or 100 for ON).

MA Volume

Object Usage: The Mixed Air Volume object (MA Volume) is a special purpose math object that calculates a mixed air volume setpoint based on the values of return air temperature, outside air temperature, and the desired volume percent between the two. Typical use is in a mixed air economizer application.

Note: Input values to the MA Volume object should be "real-world" values, otherwise erroneous output values may result. To take one extreme example, if the Return Air temperature input value is 16383 (maximum value), the Outside Air temperature input value is -163.83 (minimum value), and the Mixed Air Setpoint value is 100% (maximum value), the calculated result is -163.83. This would be the output value in a *Rev.3 or later* standard controller, or an *MN* 800 controller. In a *pre-Rev.3* standard controller, the output value would go to 0.00, instead.

Device Support: (all firmware revisions)

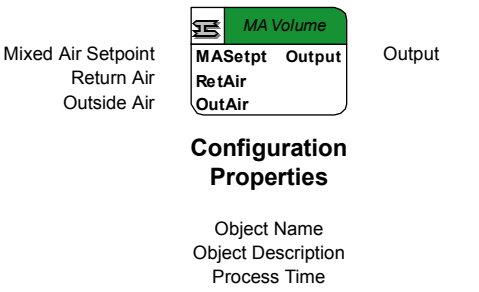
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.190 MA Volume Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.

Table–5.190 MA Volume Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.191 MA Volume Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
MASetpt	Mixed Air Setpoint	Class: Analog - The requested volume percentage of outside air in mixed air. The MA Volume object Output produces a value that maintains this volume mix as temperatures of the return air and outside air change.	0.0 to 100.0%	A not active (NA) causes the Output to be set to NA.
RetAir	Return Air	Class: Analog - The value of the return air temperature sensor, from the corresponding Analog Input object.	-163.83 to 16383	A not active (NA) causes the Output to be set to NA.
OutAir	Outside Air	Class: Analog - The value of the outside air temperature sensor, from the corresponding Analog Input object.	-163.83 to 16383	A not active (NA) causes the Output to be set to NA.

Table–5.192 MA Volume Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The calculated mixed air temperature setpoint required to generate the requested volume mix between the outside air and return air. If not active (NA), one or more of the object inputs is set to NA.	-163.8 to 16383

Applying the Object

The MA Volume object performs a special purpose calculation for use in control of a proportional mixed air economizer, based on the required values received at the three object inputs. The required values at the inputs are:

MASetpt = Mixed Air Volume Setpoint
(The desired percentage of outside air in the mixed air.)

RetAir = Return Air Temperature

OutAir = Outside Air Temperature

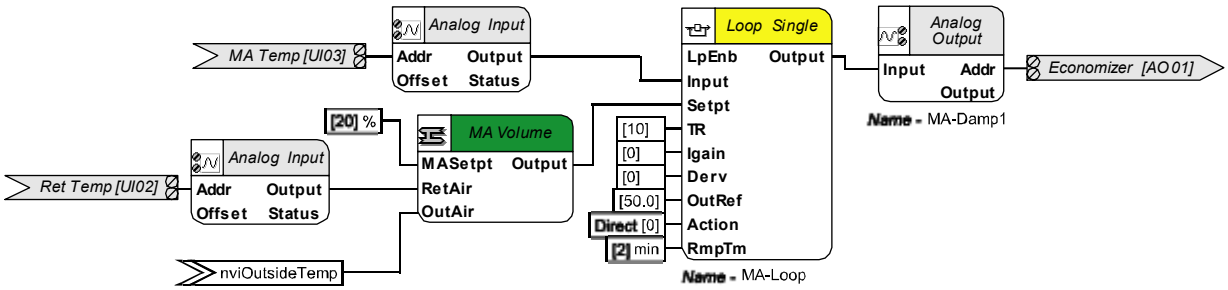
The Output of the MA Volume object is the calculated mixed air temperature setpoint required to generate the requested volume mix between the outside air and return air, using the following algorithm:

Output = RetAir - ([(RetAir - OutAir) x MASetpt] ÷ 100)

An Output of not active (NA) results if any of the inputs has a NA.

Example Application

An example MA Volume object is shown in Figure–5.92 below. The Output of the MA Volume object is the mixed air temperature setpoint for the mixed air loop in this example.



Figure–5.92 Example MA Volume Object Used in Control Logic.

Minimum Off

Object Usage: The Minimum Off object prevents the Output from being set to an ON state from an OFF state for a specified time. This ensures that any OFF period for the Output is no less than the value at the Minimum Off Time input (in minutes). The Time Enable input enables or disables the minimum off function.

The Minimum Off object is the functional opposite of the Minimum On object (page 342). Both minimum functions are available in a single object: the Dual Minimum object (page 221).

Device Support: (all firmware revisions)

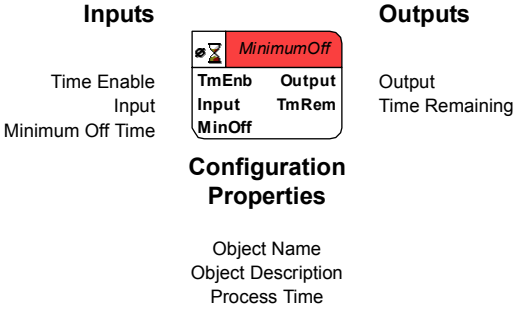
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 18 bytes (standard controller)
8 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Timer and Sequence Control

Reference Listing of All Timer Objects

Object Name	Digital Input to Digital Output Behavior
Dual Delay	Both an On Delay and an Off Delay
Dual Minimum	Both Minimum ON and Minimum OFF
Minimum On	Minimum ON period before OFF
Minimum Off	Minimum OFF period before ON
On Delay	Delay before Output ON
Off Delay	Delay before Output OFF

Properties

Table–5.193 Minimum Off Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.194 Minimum Off Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
TmEnb	Time Enable	Class: Digital - An ON or not active (NA) enables the minimum off function. An OFF at this input disables the minimum off function, causing the Output to directly follow the Input (no minimum off timeout).	—	
Input	Input	Class: Digital - The input signal to which the minimum off function is applied. An NA is evaluated as OFF.	—	See the Timing Diagram for Input to Output operation.
MinOff	Minimum Off Time	Class: Analog - The value of timeout (in minutes) for the Minimum Off period. A negative or not active (NA) value disables the Minimum Off timeout as 0.0 minutes.	0.0 to 1,000.0 minutes	Decimal values are valid. For example, 0.25 is evaluated as 15 seconds.

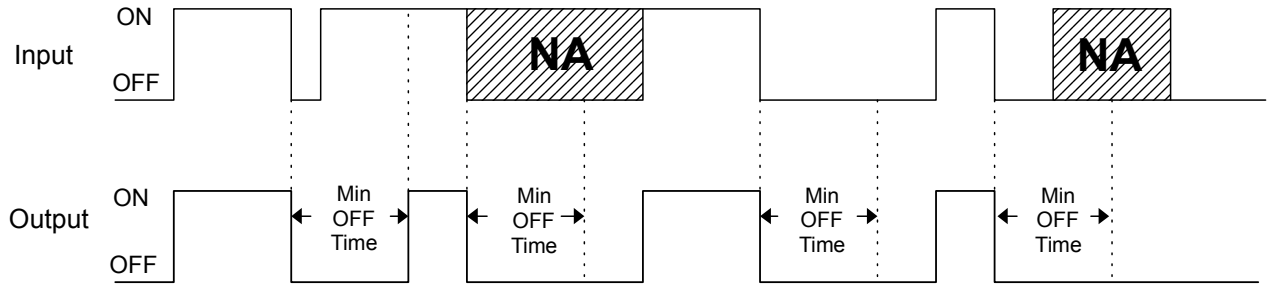
Table–5.195 Minimum Off Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The Output is set to match an Input ON state following completion of the appropriate Minimum Off timeout, or if the TmEnb input is OFF. The Output is immediately reset to OFF whenever the Input requests an OFF state.	OFF (0.0) ON (100.0)
TmRem	Time Remaining	Class: Analog - The analog value representing the amount of active Minimum Off time remaining (in whole minutes).	0 to 1,000 minutes

Applying the Object

The Minimum Off object prevents a digital output from being cycled ON without first completing an assigned OFF time (timeout), defined by the value (in minutes) at the Minimum Off Time input. This Minimum Off timeout can range from 0.0 to 1,000.0 minutes. The Time Enable input must be at ON or not active (NA) to provide the Minimum Off timeout.

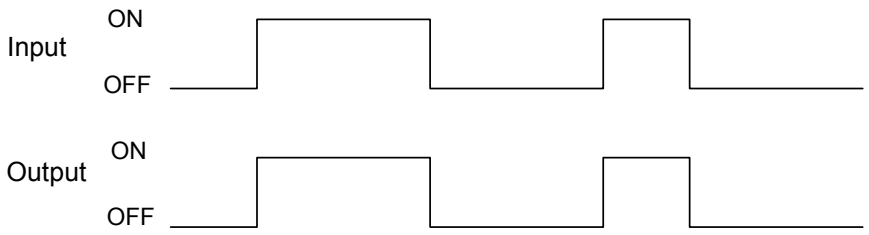
The Output is immediately set to OFF whenever an ON-to-OFF transition occurs at the Input. Figure–5.93 below shows Minimum Off object operation.



Figure–5.93 Timing Diagram for a Minimum Off Object with Timeout Enabled (Time Enable = ON or NA).

During an active Minimum Off timeout period, the Time Remaining output is the analog value for the current remaining timeout (in whole minutes). This value counts down each minute, during which time any changes to the Input are ignored and the Output remains OFF. The timeout expires at 0 (zero), allowing the Output to go to the current Input state (typically ON).

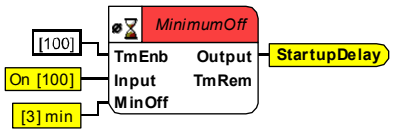
The Minimum Off function is disabled if an OFF is at the Time Enable input. In this condition, the Output directly tracks the Input Figure-5.94 and the Time Remaining output remains at 0 (zero).



Figure–5.94 Timing Diagram for a Minimum Off Object with an OFF at the Time Enable Input.

Example

One common use for the Minimum Off object is for a start-up control delay that can be used throughout an application, as needed. The figure below shows a Minimum Off object configured in this fashion.



Figure–5.95 Minimum Off Object Providing a Universal Startup Delay.

When the controller is reset via any means (power up, software reset, etc.) the output of the object remains OFF for the period of time defined for MinOff. Once this time expires, the output goes ON. If multiple resets occur in succession, such as with power “bumps”, the time period begins anew with each bump to prevent load bouncing.

The output typically is used at various “Enable” inputs and/or “selection-type” inputs (as with Select objects) as needed to enforce this purpose. In this example, the output is given the variable definition of “Startup Delay”.

Minimum On

Object Usage: The Minimum On object prevents the Output from being set to an OFF state from an ON state for a specified time. This ensures that any ON period for the Output is no less than the value at the Minimum On Time input (in minutes). The Time Enable input enables or disables the minimum On function.

The Minimum On object is the functional opposite of the Minimum Off object (page 339). Both minimum functions are available in a single object; the Dual Minimum object (page 221).

Device Support: (all firmware revisions)

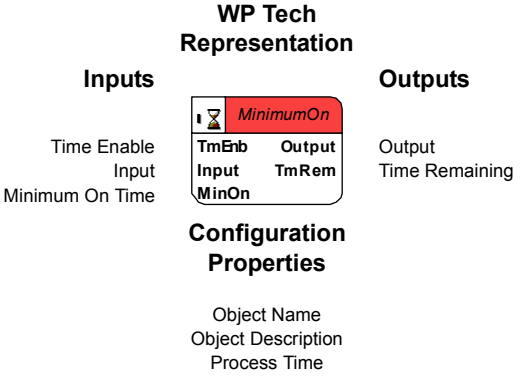
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 18 bytes (standard controller)
8 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Timer and Sequence Control

Reference Listing of All Timer Objects

Object Name	Digital Input to Digital Output Behavior
Dual Delay	Both an On Delay and an Off Delay
Dual Minimum	Both a Minimum ON and Minimum
Minimum On	Minimum ON period before OFF
Minimum Off	Minimum OFF period before ON
On Delay	Delay before Output ON
Off Delay	Delay before Output OFF

Properties

Table–5.196 Minimum On Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.197 Minimum On Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
TmEnb	Time Enable	Class: Digital - An ON or not active (NA) enables the minimum on function. An OFF at this input disables the minimum on function, causing the Output to directly follow the Input (no minimum on timeout).	—	
Input	Input	Class: Digital - The input signal to which the minimum ON function is applied. An NA is evaluated as OFF.	—	See the Timing Diagram for Input to Output operation.
MinOn	Minimum On Time	Class: Analog - The value of timeout (in minutes) for the Minimum On period. A negative or not active (NA) value disables the Minimum On timeout, acting as 0.0 minutes.	0.0 to 1,000.0 minutes	Decimal values are valid. For example, 0.25 is evaluated as 15 seconds.

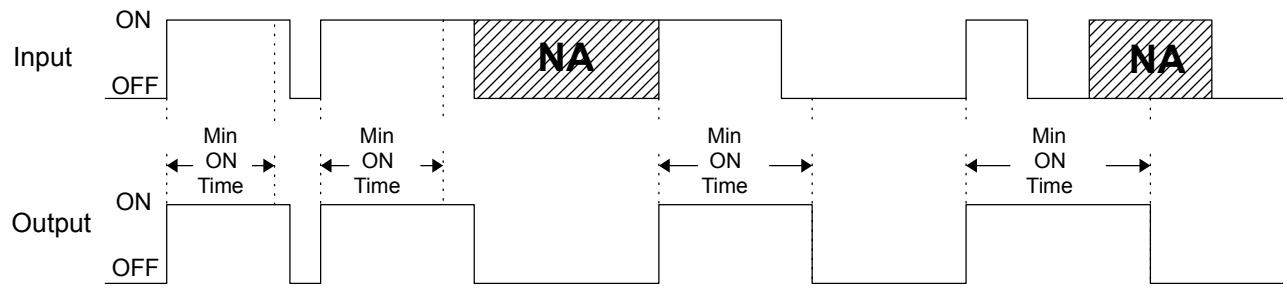
Table–5.198 Minimum On Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The Output is set to match an Input OFF state following completion of the appropriate Minimum On timeout, or if the TmEnb input is OFF. The Output is immediately reset to ON whenever the Input requests an ON state.	OFF (0.0) ON (100.0)
TmRem	Time Remaining	Class: Analog - The analog value representing the amount of active Minimum On time remaining (in whole minutes).	0 to 1,000 minutes

Applying the Object

The Minimum On object prevents a digital output from being cycled OFF without first completing an assigned ON time (timeout), defined by the value (in minutes) at the Minimum On Time input. This Minimum On timeout can range from 0.0 to 1,000.0 minutes. The Time Enable input must be at ON or not active (NA) to provide the Minimum On timeout.

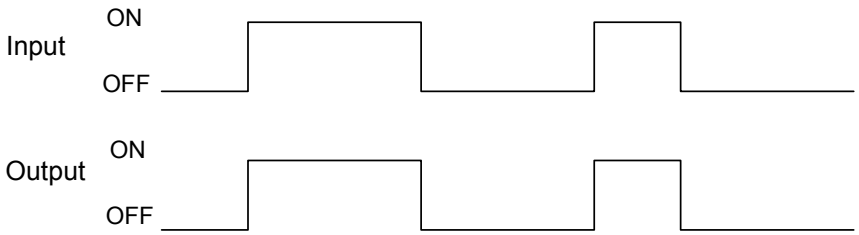
The Output is immediately set to ON whenever an OFF-to-ON transition occurs at the Input. The following Timing Diagram Figure-5.96 shows Minimum On object operation.



Figure–5.96 Timing Diagram for a Minimum On Object with Timeout Enabled (Time Enable = ON or NA).

During an active Minimum On timeout period, the Time Remaining output is the analog value for the current remaining timeout (in whole minutes). This value counts down each minute, during which time any changes to the Input are ignored and the Output remains ON. The timeout expires at 0 (zero), allowing the Output to go to the current Input state (typically OFF).

The Minimum On function is disabled if an OFF is at the Time Enable input. In this condition, the Output directly tracks the Input Figure-5.97 and the Time Remaining output remains at 0 (zero).



Figure–5.97 Timing Diagram for a Minimum On object with an OFF at the Time Enable Input.

Note: After a controller reset the object operates as if the input and output were off prior to the reset.

Momentary Start / Stop

Object Usage: The Momentary Start / Stop object is a point-type object that provides timed start and stop pulses to the two physical (hardware) outputs as well as to the two digital outputs. Uses include start and stop pulses for motor control or ON and OFF pulses for other uses, such as long delays and timed overrides without reset.

Note: A controller reset results in a Start Pulse or Stop Pulse, depending on the present valid state (ON or OFF) at the Input. A not active (NA) input is ignored until a valid Input state is detected.

Device Support: (all firmware revisions)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4

MNL-11Rxx, -13Rxx
where xx = F2 or F3

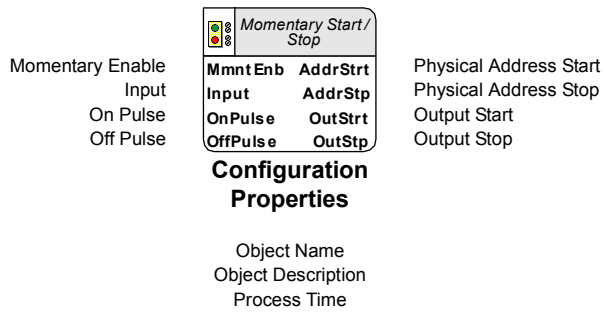
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 16 bytes
RAM: 26 bytes (standard controller)
10 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.199 Momentary Start / Stop Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.200 Momentary Start / Stop Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
MmntEnb	Momentary Enable	Class: Digital - An ON or not active (NA) enables the Momentary Start / Stop function, where all Input changes are processed. An OFF at this input disables the Momentary Start / Stop function, with the Input evaluated as a constant OFF. Also, an ON-to-OFF transition causes a stop pulse sequence on stop outputs (AddrStp and OutStp) for the duration defined by the Off Pulse value.	—	Note that all standard rules apply, where the stop pulse sequence cannot occur until the completion of any start pulse sequence (if one is in progress).
Input	Input	Class: Digital - The input signal to which the Momentary Start / Stop function is applied. • A transition to ON causes a start pulse sequence on start outputs (AddrSP and OutStrt) for the duration defined by the On Pulse value. • A transition to OFF causes a stop pulse sequence on stop outputs (AddrStp and OutStp) for the duration defined by the Off Pulse value.	—	A transition to not active (NA) causes no change to the outputs. See the Timing Diagram for Input to Output operation.
OnPulse	On Pulse	Class: Analog - The on pulse duration (in seconds) for a start pulse sequence. A negative or NA value acts as 0.0 seconds (no start pulse).	0.1 to 10,000.0 seconds	Values between 0.01 and 0.09 are defaulted to 0.10 seconds.
OffPulse	Off Pulse	Class: Analog - The off pulse duration (in seconds) for a stop pulse sequence. A negative or NA value acts as 0.0 seconds (no stop pulse).	0.1 to 10,000.0 seconds	

Table–5.201 Momentary Start / Stop Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
AddrStrt	Physical Address Start	Class: Analog - The physical hardware address (digital output terminal point on the controller) assigned to the momentary start / stop (Start) pulse function.	Dependent on the controller platform selected.
AddrStp	Physical Address Start	Class: Analog - The physical hardware address (digital output terminal point on the controller) assigned to the momentary start / stop (Stop) pulse function.	Dependent on the controller platform selected.
OutStrt	Output Start	Class: Digital - Toggles from OFF-to-ON when a transition from OFF-to-ON occurs at the Input. This output remains ON for the duration (in seconds) defined by the On Pulse value, returning afterwards to OFF.	OFF (0.0) ON (100.0)
OutStp	Output Stop	Class: Digital - Toggles from OFF-to-ON when a transition from ON-to-OFF occurs at the Input. This output remains ON for the duration (in seconds) defined by the OFF Pulse value, returning afterwards to OFF.	OFF (0.0) ON (100.0)

Applying the Object

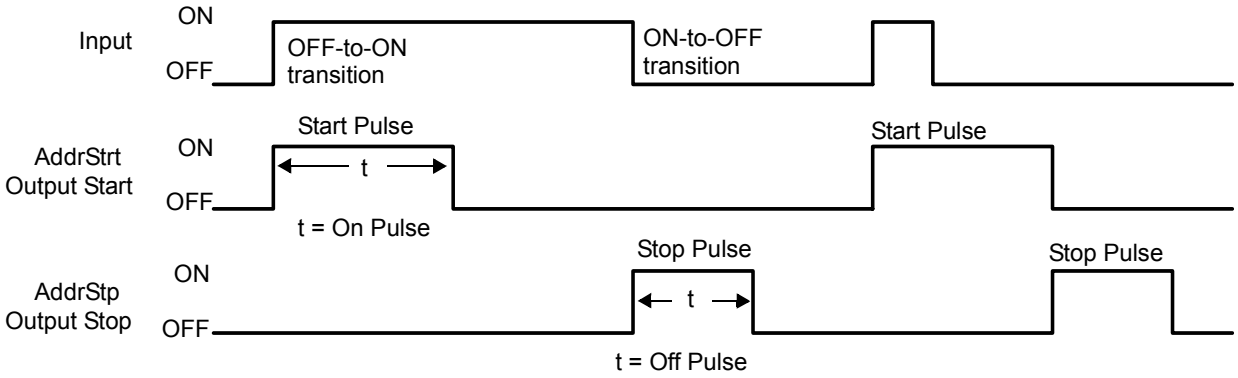
The Momentary Start / Stop object allows start / stop control of equipment requiring a binary pulse rather than maintained contact for changing states. This point-type object provides two physical output addresses; one for a Start pulse and the other for a Stop pulse. The object also includes two logical digital outputs (Output Start and Output Stop) that indicate the present Start pulse and Stop pulse states (OFF or ON). These outputs can also be used for long delays or timed overrides without reset.

Values at the inputs On Pulse and Off Pulse define the time duration (in seconds) for a start pulse and a stop pulse, respectively. Pulse durations can be from a minimum of 0.1 to a maximum of 10,000 seconds. Values between 0.01 and 0.09 seconds are defaulted to 0.10 seconds.

Start Pulse and Stop Pulse

A Start Pulse results from an OFF-to-ON (or NA-to-ON) transition at the Input. During a Start Pulse sequence, the physical AddrStrt and Output Start outputs remain ON for the duration defined by the value at the On Pulse input. A negative value or NA at On Pulse disables the start-pulse sequence and outputs.

A Stop Pulse results from an ON-to-OFF (or NA-to-OFF) transition at the Input. During a Stop Pulse sequence, the physical AddrStp and Output Stop outputs remain ON for the duration defined by the value at the Off Pulse input. A negative value or NA at OFF Pulse disables the stop-pulse sequence and outputs.



Figure–5.98 Momentary Start / Stop Object Timing Diagram.

Pulse in Progress

While either a start or stop pulse sequence is active, changes at the Input are ignored. The Input is monitored only after completion of the present pulse sequence. This prevents the output pulses from generating short cycle start and stop sequences.

Momentary Enable

When an ON-to-OFF (or NA-to-OFF) transition occurs at the Momentary Enable input, the Momentary Start / Stop object produces a stop pulse sequence (as described previously). As long as the Momentary Enable input remains in an OFF state, the Input is considered as OFF. Note that all standard rules apply where the Stop Pulse sequence can not occur until the completion of a Start Pulse sequence (if in progress).

Mul / Add

Object Usage: The Mul / Add object is a three-input math object for use with analog values (AV). This object multiplies Input[1] and Input[2] and then adds Input[3].

Output = (AV1 x AV2) + AV3

A minimum of two valid inputs are required to produce a valid output.

Device Support: (all firmware revisions)

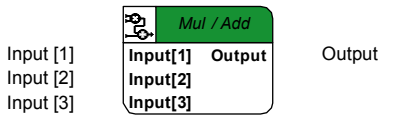
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.202 Mul / Add Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.203 Mul / Add Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The first value or multiplicand in the equation. (**AV1** x AV2) + AV3	-163.83 to 16383	If not active (NA) is present at both Inputs[1] and [2], the output is set to NA.
Input[2]	Input [2]	Class: Analog - The second value used as the multiplier. (AV1 x **AV2**) + AV3	-163.83 to 16383	
Input[3]	Input [3]	Class: Analog - The third value added to the previous result. (AV1 x AV2) + **AV3**	-163.83 to 16383	If not active (NA), the output is set to NA.

Table–5.204 Mul / Add Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The result of the equation: (AV1 x AV2) + AV3. Refer to Truth Table for effects of input(s) with not active (NA).	-163.83 to 16383

Applying the Object

The Mul / Add object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. The equation specific to the Mul / Add object is:

(AV1 x AV2) + AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object.
A truth table Figure-5.42 shows how NA inputs affect the output of the Mul / Add object.

Table–5.205 Mul / Add Object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	(AV1 x AV2) + AV3
AV1	AV2	NA	NA
AV1	NA	AV3	AV1 + AV3
NA	AV2	AV3	AV2 + AV3
NA	NA	NA	NA
NA	NA	AV3	NA

Mul / Div

Object Usage: The Mul / Div object is a three-input math object for use with analog values (AV). This object multiplies Input[1] and Input[2] and then divides the result by Input[3].

Output = (AV1 x AV2) ÷ AV3

A minimum of two valid inputs are required to produce a valid output.

Device Support: (all firmware revisions)

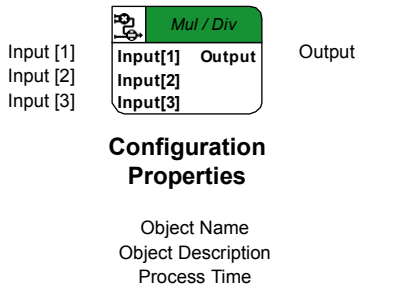
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.206 Mul / Div Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.207 Mul / Div Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The first value or multiplicand in the equation. (**AV1** x AV2) ÷ AV3	-163.83 to 16383	If not active (NA) is present at both Inputs[1] and [2], the output is set to NA.
Input[2]	Input [2]	Class: Analog - The second value used as the multiplier to the first value. (AV1 x **AV2**) ÷ AV3	-163.83 to 16383	
Input[3]	Input [3]	Class: Analog - The third value or divisor, which is divided into the previous result. (AV1 x AV2) ÷ **AV3**	-163.83 to 16383	If not active (NA), the output is set to NA.

Table–5.208 Mul / Div Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The result of the equation: (AV1 x AV2) ÷ AV3 . Refer to Truth Table for effects of input(s) with not active (NA).	-163.83 to 16383

Applying the Object

The Mul / Div object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. The equation specific to the Mul / Div object is:

(AV1 x AV2) ÷ AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Note: A value of zero at Input[3] causes the output to process a “divide by zero” that sets the output to either a minimum (-163.83) or a maximum (16383) value based upon the results of the first two inputs. A negative result causes the output to be set to the minimum (-163.83) value. A positive result causes the output to be set to the maximum (16383) value.

- Result < 0 sets the output to the minimum (-163.83) value.
- Result > 0 sets the output to the maximum (16383) value.

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. A truth table Figure-5.209 shows how NA inputs affect the output of the Mul / Div object.

Table–5.209 Mul / Div Object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	(AV1 x AV2) ÷ AV3
AV1	AV2	NA	NA
AV1	NA	AV3	AV1 ÷ AV3
NA	AV2	AV3	AV2 ÷ AV3
NA	NA	NA	NA
NA	NA	AV3	NA

Off Delay

Object Usage: The Off Delay object monitors a digital Input and provides a delayed digital Output response to an ON-to-OFF transition. Delay times can be specified from 0.0 to 1,000.0 minutes. A Time Remaining output provides the current remaining minutes in any active Off Delay. Any OFF-to-ON transition at the Input is always directly tracked at the Output. The Off Delay function can be disabled with an OFF at the Time Enable input, which causes the Output to directly track the Input state. A not active (NA) to the Input is evaluated as an OFF.

The Off Delay object provides the opposite function of an On Delay object (page 355). Both the Off delay and On delay functions are available in a single object: the Dual Delay object (page 218).

Digital Output = Digital Input (following the off delay)

Device Support: (all firmware revisions)

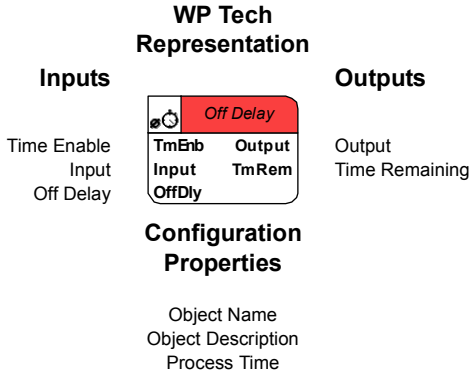
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 18 bytes (standard controller)
8 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Timer and Sequence Control

Reference Listing of All Timer Objects

Object Name	Digital Input to Digital Output Behavior
Dual Delay	Both an On Delay and an Off Delay
Dual Minimum	Both Minimum ON and Minimum OFF
Minimum On	Minimum ON period before OFF
Minimum Off	Minimum OFF period before ON
On Delay	Delay before Output ON
Off Delay	Delay before Output OFF

Properties

Table–5.210 Off Delay Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.211 Off Delay object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
TmEnb	Time Enable	Class: Digital - An ON or not active (NA) enables the Off Delay function. An OFF at this input disables the Off Delay function, causing the Output to directly follow the Input (no delay).	—	
Input	Input	Class: Digital - The input signal to which the Off Delay is applied. An NA is evaluated as OFF.	—	See the Timing Diagram for Input to Output operation.
OffDly	Off Delay	Class: Analog - The value of the OFF delay time in minutes. A negative or not active (NA) value disables the delay as 0.0 minutes.	0.0 to 1,000.0 minutes	Decimal values are valid. For example, 0.25 is evaluated as 15 seconds.

Table–5.212 Off Delay object Output Properties.

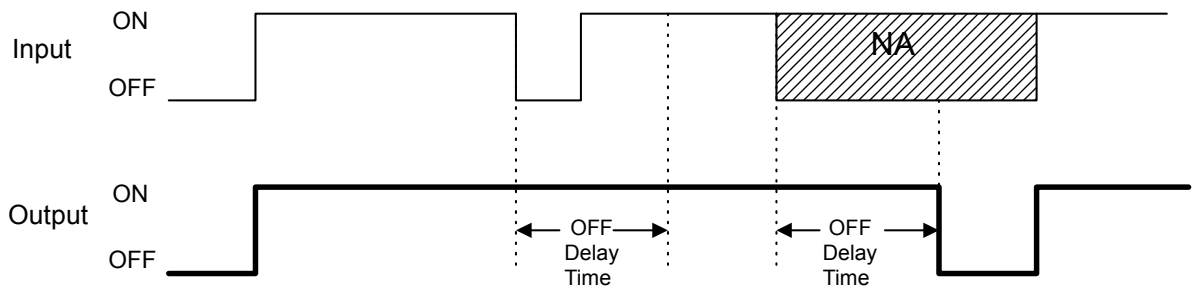
Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The Output is set to OFF following completion of the Off Delay time, or directly tracks the Input if the TmEnb input is OFF.	OFF (0.0) ON (100.0)
TmRem	Time Remaining	Class: Analog -The analog value representing the amount of active OFF delay time (in whole minutes).	0 to 1,000 minutes

Applying the Object

The Off Delay object provides a delayed output response to an ON-to-OFF transition. The Off Delay object can be used in applications requiring short-cycle control protection or purge control sequences. The Off Delay object acts as the functional opposite of the On Delay object. Both the Off Delay function and the On Delay function are also available in a single object: the Dual Delay object (page 218).

An Off Delay is triggered by an ON-to-OFF transition received as a digital signal on the Off Delay object's Input. An Off Delay can lasts from 0.1 to 1000.0 minutes, based on the value present at the input Off Delay. Any OFF-to-ON transition at the Input is always directly tracked at the Output.

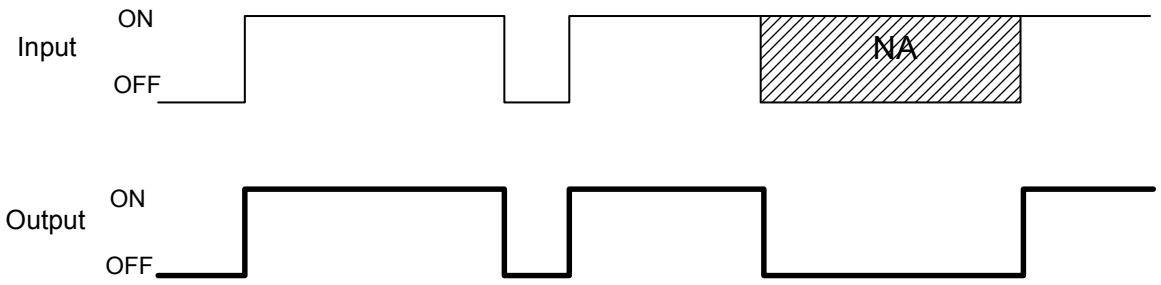
The Off Delay object requires the Time Enable input to be either ON or not active (NA) to provide the Off Delay function. Figure–5.99 below is a timing diagram showing the operation of an Off Delay object.



Figure–5.99 Timing Diagram for a Off Delay object with the Delay Function Enabled (Time Enable input is ON or NA).

During an active Off Delay, the Time Remaining output is an analog value for the current remaining delay time (in whole minutes). This value counts down each minute as the delay timer expires, where the Time Remaining output is at 0 (zero) and the Output is free to go OFF (if the Input is still OFF).

The Off Delay function is disabled while an OFF is at the Time Enable input. In this condition, the Output directly tracks the Input without delay Figure-5.100, and the Time Remaining output remains at 0 (zero).



Figure–5.100 Timing Diagram for a Off Delay Object with the Delay Function Disabled (Time Enable input is OFF).

Note: After a controller reset the object operates as if the input and output were off prior to the reset.

On Delay

Object Usage: The On Delay object monitors a digital Input and provides a delayed digital Output response to an OFF-to-ON transition. Delay times can be specified from 0.0 to 1,000.0 minutes.
A Time Remaining output provides the current remaining minutes in any active On Delay.
Any ON-to-OFF transition at the Input is always directly tracked at the Output. The On Delay function can be disabled with an OFF at the Time Enable input, which causes the Output to directly track the Input state. A not active (NA) to the Input is evaluated as an OFF.

The On Delay object provides the opposite function of an Off Delay object (page 352). Both the On delay and Off delay functions are available in a single object: the Dual Delay object (page 218).

Digital Output = Digital Input (following the on delay)

Device Support: (all firmware revisions)

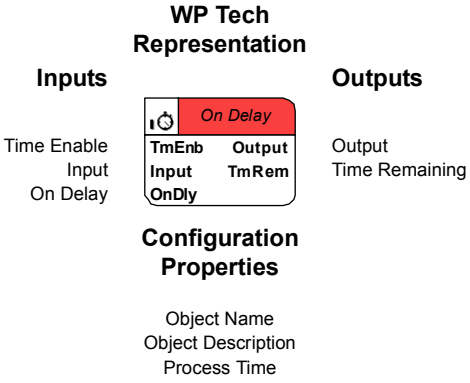
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 18 bytes (standard controller)
8 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Timer and Sequence Control

Reference Listing of All Timer Objects

Object Name	Digital Input to Digital Output Behavior
Dual Delay	Both an On Delay and an Off Delay
Dual Minimum	Both Minimum ON and Minimum OFF
Minimum On	Minimum ON period before OFF
Minimum Off	Minimum OFF period before ON
On Delay	Delay before Output ON
Off Delay	Delay before Output OFF

Properties

Table–5.213 On Delay Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.214 On Delay Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
TmEnb	Time Enable	Class: Digital - An ON or not active (NA) enables the On Delay function. An OFF at this input disables the On Delay function, causing the Output to directly follow the Input (no delays).	—	
Input	Input	Class: Digital - The input signal to which the On Delay is applied. An NA is evaluated as OFF.	—	See the Timing Diagram for Input to Output operation.
OnDly	On Delay	Class: Analog - The value of the ON delay time in minutes. A negative or not active (NA) value disables the delay as 0.0 minutes.	0.0 to 1,000.0 minutes	Decimal values are valid. For example, 0.25 is evaluated as 15 seconds.

Table–5.215 On Delay Object Output Properties.

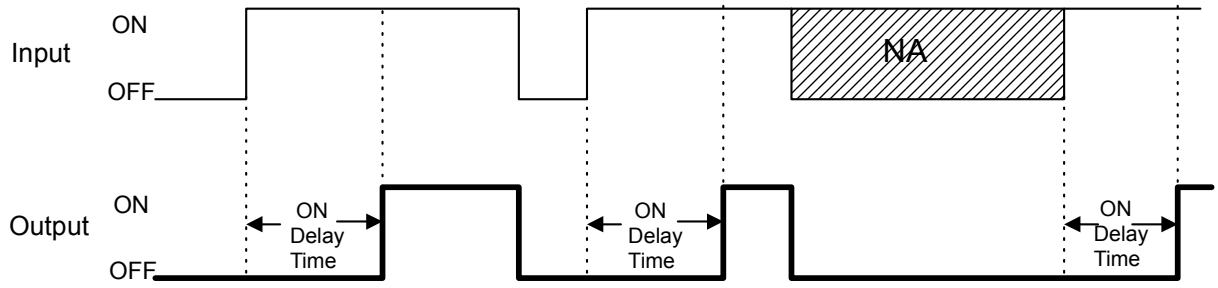
Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The Output is set to ON following completion of the On Delay time, or directly tracks the Input if the TmEnb input is OFF.	OFF (0.0) ON (100.0)
TmRem	Time Remaining	Class: Analog - The analog value representing the amount of active On Delay time (in whole minutes).	0 to 1,000 minutes

Applying the Object

The On Delay object provides a delayed output response to an OFF-to-ON transition. The ON delay object can be used in applications requiring short cycle control protection or delayed start control sequences. The ON delay object acts as the functional opposite of the Off Delay object. Both the On Delay function and the Off Delay function are also available in a single object: the Dual Delay object (page 218).

An On Delay is triggered by an OFF-to-ON transition received as a digital signal on the On Delay object's Input. An On Delay can lasts from 0.1 to 1000.0 minutes, based on the value present at the input On Delay. Any ON-to-OFF transition at the Input is always directly tracked at the Output.

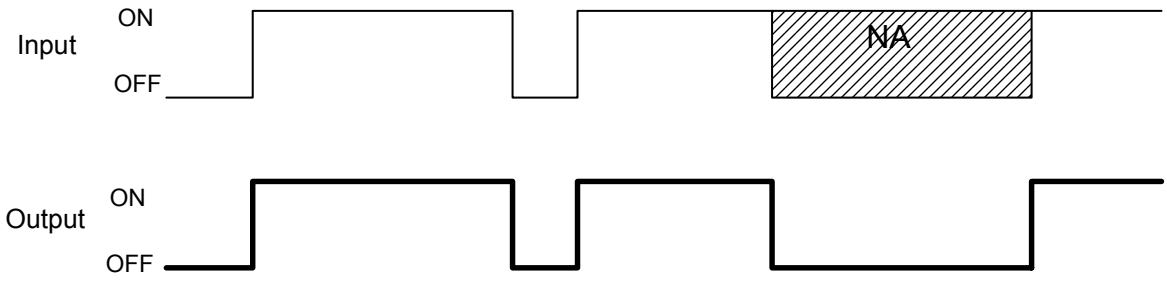
The On Delay object requires the Time Enable input to be either ON or not active (NA) to provide the On Delay function. Figure–5.101 is a timing diagram that shows the operation of an On Delay object.



Figure–5.101 Timing Diagram for a On Delay object with the Delay Function Enabled (Time Enable input is ON or NA).

During an active On Delay, the Time Remaining output is an analog value for the current remaining delay time (in whole minutes). This value counts down each minute as the delay timer expires, where the Time Remaining output is at 0 (zero) and the Output is free to go ON (if the Input is still ON).

The On Delay function is disabled while an OFF is at the Time Enable input. In this condition, the Output directly tracks the Input without delay Figure-5.102, and the Time Remaining output remains at 0 (zero).



Figure–5.102 Timing Diagram for an On Delay Object with the Delay Function Disabled (Time Enable input is OFF).

OR / AND

Object Usage: The OR / AND object is a three-input logic object for use with OFF and ON digital values (DV). The output of the object is a digital ON when either Input[1] or Input[2] is in a digital ON state and Input[3] is also in a digital ON state.
An unconnected input is considered invalid or not active (NA), and is ignored in the object's algorithm. If all inputs are NA, the output is set to NA.

Output = (In1 OR In2) AND In3

Device Support: (all firmware revisions)

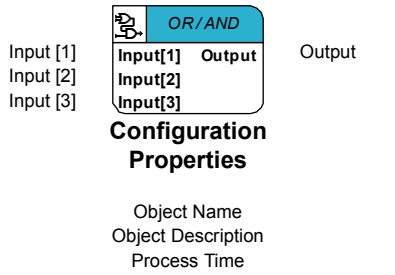
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

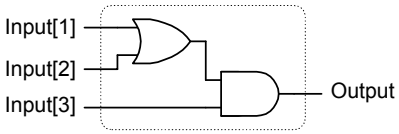
WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

Logic



OR / AND

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Digital Logic Objects

Object Name	Digital Object Algorithm (all are three-input unless noted)
AND / AND	In1 AND In2 AND In3
AND / OR	(In1 AND In2) OR In3
Clocked SR	Clocked Set-Reset Flip-Flop Logic
EXOR	Two-input, Exclusive OR
Latch	Digital Sample and Hold or Latch
OR / AND	(In1 OR In2) AND In3
OR / OR	In1 OR In2 OR In3
SR Flip-Flop	Two-input, Set-Reset Flip-Flop Logic

Properties

Table–5.216 OR / AND Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.217 OR / AND Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Digital - The first input evaluated for an ON. If ON, the third input is evaluated. A not active (NA) is ignored. **In1** OR In2 AND In3	—	See the Truth Table for all possible input combinations and corresponding outputs.
Input[2]	Input [2]	Class: Digital - The second input evaluated for an ON. If ON, the third input is evaluated. A not active (NA) is ignored. In1 OR **In2** AND In3	—	See the Truth Table for all possible input combinations and corresponding outputs.
Input[3]	Input [3]	Class: Digital - The third input evaluated for an ON. If OFF, the output is set to OFF. If not active (NA), the input is ignored unless all inputs are NA, in which case the output is also set to NA. In1 OR In2 AND **In3**	—	See the Truth Table for all possible input combinations and corresponding outputs.

Table–5.218 OR / AND Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The output indicates the result of the logic algorithm. If not active (NA) is present at all three inputs, the output is set to NA.	OFF (0.0) ON (100.0)

Applying the Object

The OR / AND object is similar to other three-input logic objects, which also process OFF and ON digital values (DV) and produce an DV output. The object's algorithm use this logic:

(In1 OR In2) AND In3

Assuming valid input values, Input[3] must be ON with at least one of the first two inputs Input[1] and Input[2] also ON before the Output is set to ON, otherwise the Output is OFF. If an Input is not active (NA) it is invalid (ignored), allowing less than three Inputs to be evaluated for an ON state. If all Inputs are NA, the Output is set to NA. The following truth table provides all OR / AND object combinations:

Table–5.219 Truth Table for OR / AND Object.

Input[1]	Input[2]	Input[3]	Output
OFF	OFF	OFF	OFF
OFF	OFF	ON	OFF
OFF	OFF	NA	OFF
OFF	ON	OFF	OFF
OFF	ON	ON	ON
OFF	ON	NA	ON
OFF	NA	OFF	OFF
OFF	NA	ON	OFF
OFF	NA	NA	OFF
ON	OFF	OFF	OFF
ON	OFF	ON	ON
ON	OFF	NA	ON
ON	ON	OFF	OFF
ON	ON	ON	ON
ON	ON	NA	ON
ON	NA	OFF	OFF
ON	NA	ON	ON
ON	NA	NA	ON
NA	OFF	OFF	OFF
NA	OFF	ON	OFF
NA	OFF	NA	OFF
NA	ON	OFF	OFF
NA	ON	ON	ON
NA	ON	NA	ON
NA	NA	OFF	OFF
NA	NA	ON	ON
NA	NA	NA	NA

As with other logic objects, Inputs to this object are typically digital values from a connection to another object's output, represented numerically with a 0 (OFF) or 100 (ON). However, Inputs can also process analog values or a value from an assigned constant. An Input with an analog value or constant is evaluated as OFF if the value is negative or equal to zero (0.0), or ON if the Input has any positive value greater than zero.

OR / OR

Object Usage: The OR / OR object is a three-input logic object for use with OFF and ON digital values (DV). The output of the object is a digital ON whenever any of the three inputs is in a digital ON state. An unconnected input is considered invalid or not active (NA), and is ignored in the object's algorithm. If all inputs are NA, the output is set to NA.

Output = In1 OR In2 OR In3

Device Support: (all firmware revisions)

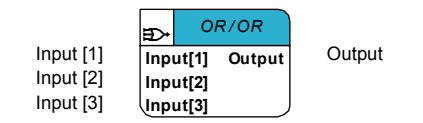
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

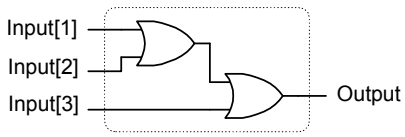
WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

Logic



OR / OR

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Digital Logic Objects

Object Name	Digital Object Algorithm (all are three-input unless noted)
AND / AND	In1 AND In2 AND In3
AND / OR	(In1 AND In2) OR In3
Clocked SR	Clocked Set-Reset Flip-Flop Logic
EXOR	Two-input, Exclusive OR
Latch	Digital Sample and Hold or Latch
OR / AND	(In1 OR In2) AND In3
OR / OR	In1 OR In2 OR In3
SR Flip-Flop	Two-input, Set-Reset Flip-Flop Logic

Properties

Table–5.220 OR / OR Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.221 OR / OR Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Digital - The first input evaluated for an ON. If OFF, the second input is evaluated. A not active (NA) is ignored. **In1** OR In2 OR In3	—	See the Truth Table for all possible input combinations and corresponding outputs.
Input[2]	Input [2]	Class: Digital - The second input evaluated for an ON. If OFF, the third input is evaluated. A not active (NA) is ignored. In1 OR **In2** OR In3	—	See the Truth Table for all possible input combinations and corresponding outputs.
Input[3]	Input [3]	Class: Digital - The third input evaluated for an ON. If OFF, the output is set to OFF unless either Input[1] or [2] are ON. If all inputs are NA, the output is set to NA. In1 OR In2 OR **In3**	—	See the Truth Table for all possible input combinations and corresponding outputs.

Table–5.222 OR / OR Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The output indicates the result of the logic algorithm. If not active (NA) is present at all three inputs, the output is set to NA.	OFF (0.0) ON (100.0)

Applying the Object

The OR / OR object is similar to other three-input logic objects, which also process OFF and ON digital values (DV) and produce an DV output. The object's algorithm is unique in the use of two logical OR operators:

In1 OR In2 OR In3

This logic provides an Output of ON whenever any of the inputs is ON, and is OFF only when the three inputs (Inputs[1], [2], and [3]) are all OFF.

An unconnected Input is considered not active (NA) and is invalid (ignored), allowing less than three Inputs to be evaluated. If all Inputs are NA, the Output is set to NA. The following truth table Table–5.223 provides all OR / OR object input/output combinations.

Table–5.223 Truth Table for OR / OR Object.

Input[1]	Input[2]	Input[3]	Output
OFF	OFF	OFF	OFF
OFF	OFF	ON	ON
OFF	OFF	NA	OFF
OFF	ON	OFF	ON
OFF	ON	ON	ON
OFF	ON	NA	ON
OFF	NA	OFF	OFF
OFF	NA	ON	ON
OFF	NA	NA	OFF
ON	OFF	OFF	ON
ON	OFF	ON	ON
ON	OFF	NA	ON
ON	ON	OFF	ON
ON	ON	ON	ON
ON	ON	NA	ON
ON	NA	OFF	ON
ON	NA	ON	ON
ON	NA	NA	ON
NA	OFF	OFF	OFF
NA	OFF	ON	ON
NA	OFF	NA	OFF
NA	ON	OFF	ON
NA	ON	ON	ON
NA	ON	NA	ON
NA	NA	OFF	OFF
NA	NA	ON	ON
NA	NA	NA	NA

As with other logic objects, Inputs to this object are typically digital values from a connection to another object's output, represented numerically with a 0 (OFF) or 100 (ON). However, Inputs can also process analog values or a value from an assigned constant. An Input with an analog value or constant is evaluated as OFF if the value is equal to or less than zero (0.0), or ON if the Input has any positive value greater than zero.

OSS

Object Usage: The Optimum Start Stop (OSS) object is applied to systems such as air handlers, boilers, and other controlled devices that operate in both occupied and unoccupied modes. The optimum start function is designed to achieve occupied setpoint comfort levels with the least amount of energy usage. The optimum stop function is designed to use the least amount of energy while maintaining a specified comfort level during the transition from an occupied period to a scheduled unoccupied event.

Device Support:

MN 800 series

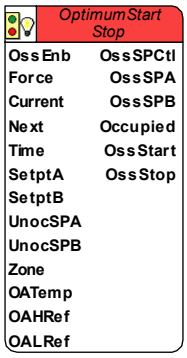
Memory Requirements: (per object)

EEPROM: 52 bytes
RAM: 82 bytes

WP Tech Representation

Inputs

OSS Enable
Force Occupied
Current
Next
Time
Setpoint A
Setpoint B
Unoccupied Setpoint A
Unoccupied Setpoint B
Zone Temperature
Outside Air Temperature
Outside Air High Reference
Outside Air Low Reference



Outputs

OSS Setpoint Control
OSS Setpoint A
OSS Setpoint B
Occupied
OSS Start Status
OSS Stop Status

See Additional Output Properties for additional outputs.

Configuration Properties

Object Name
Object Description
OSS Mode
Input Select
Zone Cooling Factor
Zone Heating Factor
Outside Air Cooling Factor
Outside Air Heating Factor
Coast Factor
Comfort zone
Cooling Start Limit
Heating Start Limit
Stop Limit

WP Tech Stencil:
Schedule Control

Properties

Table–5.224 OSS Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
OssMode	OSS Mode	Class: Analog - Selects either zone temperature alone, or both zone and outside air temperature, for use by the Optimum Start / Stop algorithm to calculate optimum start and stop times.	0	Temperature value used for calculation: 0 - Zone 1 - Outside air	A not active (NA) or values outside the defined range cause the selection to default to Zone.

Table–5.224 OSS Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
InSel	Input Select	Class: Analog - Defines the data format of the Current and Next inputs. 0 - Digital, data is interpreted digitally 1 - SNVT_occupancy, data is interpreted as an enumerated value, following the definition of SNVT_occupancy	0	How algorithm interprets Current and Next input data: 0 - Digital 1 - SNVT_occupancy	An Input Select that is not active (NA) or out of range causes the algorithm to default to an Input Select of (0).
ZnClFct	Zone Cooling Factor	Class: Analog - A constant that is used when calculating the zone temperature's influence on the optimum start time for cooling conditions. This value is defined in minutes per degree and is applied by the algorithm as required based upon the OSS Mode selection (*Zone or Outside Air*).	NA	0 to 1000	A not active (NA), zero, or negative value causes the cooling optimum start function to be disabled.
ZnHtFct	Zone Heating Factor	Class: Analog - A constant that is used when calculating the zone temperature's influence on the optimum start time for heating conditions. This value is defined in minutes per degree and is applied by the algorithm as required based upon the OSS Mode selection (*Zone or Outside Air*).	NA	0 to 1000	A not active (NA), zero, or negative value causes the heating optimum start function to be disabled.
OAClFct	Outside Air Cooling Factor	Class: Analog - A constant that is used when calculating the outside air's influence on the optimum start time for cooling conditions. This value is defined in minutes per degree and is only applied by the algorithm when using an OSS Mode selection of *Outside Air*.	NA	0 to 1000	A not active (NA), zero, or negative value causes the cooling optimum start function to be disabled. The algorithm will ignore the Outside Air Cooling Factor when OSS Mode selection is *Zone*.
OAHtFct	Outside Air Heating Factor	Class: Analog - A constant that is used when calculating the outside air's influence on the optimum start time for heating conditions. This value is defined in minutes per degree and is only applied by the algorithm when using an OSS Mode selection of *Outside Air*.	NA	0 to 1000	A not active (NA), zero, or negative value causes the heating optimum start function to be disabled. The algorithm will ignore the Outside Air Heating Factor when OSS Mode selection is *Zone*.

Table–5.224 OSS Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
CstFctr	Coast Factor	Class: Analog - A constant that is used when calculating the optimum stop time when OSS Mode is set for *Outside Air* operation. This value is defined in minutes per degree and is applied by the algorithm as required.	NA	0 to 1000	A not active (NA), zero, or negative value causes the optimum stop function to be disabled. The algorithm will ignore the Coast Factor when OSS Mode selection is *Zone*.
Comfort	Comfort Zone	Class: Analog - Used by the algorithm in the calculation of cooling / heating comfort targets during the optimum start sequence. The comfort zone value is also applied to the occupied heating and cooling setpoints to generate the coast setpoints during optimum stop conditions.	NA	0 to 1000	A not active (NA) or negative value causes Comfort Zone to default to a value of zero.
ClStrLmt	Cooling Start Limit	Class: Analog - Defines the maximum number of minutes before scheduled occupancy time, during which optimum start cooling is allowed to occur.	NA	0 to 1000	A not active (NA), zero, or negative value causes the optimum start function to be disabled.
HtStrLmt	Heating Start Limit	Class: Analog - Defines the maximum number of minutes before scheduled occupancy time, during which optimum start heating is allowed to occur.	NA	0 to 1000	A not active (NA), zero, or negative value causes the heating optimum start function to be disabled.
StpLmt	Stop Limit	Class: Analog - Defines the maximum number of minutes before scheduled unoccupied time, during which optimum stop is allowed to occur.	NA	0 to 1000	A not active (NA), zero, or negative value causes the cooling optimum stop function to be disabled.

Table–5.225 OSS Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
OssEnb	OSS Enable	Class: Digital - Enables and disables the optimum start / stop functions. Digital ON - Enables optimum start / stop functions Digital OFF - Disables optimum start / stop functions	—	Disabling the optimum start / stop functions (Digital OFF) causes the control logic outputs and setpoint output values to follow the “Current” event and/or the Force Occupied input condition, without optimum start / stop intervention. A not active (NA) causes the OSS Enable to default to Digital ON enabling the optimum start / stop functions.
Force	Force Occupied	Class: Digital - When ON, forces the control logic outputs (OssSPCtl and Occupied) to indicate occupied mode (Digital ON), regardless of active (prestart / prestop) or inactive OSS conditions. Also, the OssStart and OssStop outputs are set to Digital OFF, and the OssSPA and OssSPB values will be set to the Setpoint A and Setpoint B values, respectively. Digital ON - Disables the optimum start / stop functions and forces the control logic outputs to indicate occupied mode. Digital OFF - causes the Force Occupied function to be inactive.	—	A not active (NA) causes the Force Occupied function to be inactive.

Table–5.225 OSS Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
Current	Current	Class: Analog - Provides the currently active event value. With an *Input Select of "Digital"*, the algorithm interprets the Current input digitally, where: value = negative or zero = unoccupied (Digital OFF) value > zero = occupied (Digital ON). value = not active (NA) = algorithm defaults to an occupied condition, disabling the optimum start / stop function. In this case, the control logic outputs (OssSPCtl and Occupied) are set to indicate occupied mode (Digital ON), the OssStart and OssStop outputs are set to Digital OFF, and the OssSPA and OssSPB values are set to the Setpoint A and Setpoint B values, respectively. With an *Input Select of "Enumeration"*, the algorithm interprets the Current input data as an enumerated value that follows the SNVT_occupancy definition: 0 = Occupied 1 = Unoccupied 2 = Bypass 3 = Standby 255 = Null A Current input value of Bypass (2), Standby (3), or Null (255), or any value outside the defined enumeration range causes the algorithm to default to an occupied condition, disabling the optimum start / stop function. In this case, the control logic outputs (OssSPCtl and Occupied) are set to indicate occupied mode (Digital ON), the OssStart and OssStop outputs are set to Digital OFF, and the OssSPA and OssSPB values will be set to the Setpoint A and Setpoint B values, respectively.	Based upon the Input Select configuration property	In cases where both Current and Next input values are the same, either both occupied or both unoccupied, the algorithm defaults to the Current event and disables the optimum start / stop function.

Table–5.225 OSS Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
Next	Next	Class: Analog - Provides the next scheduled event value. With an *Input Select of "Digital"*, the algorithm interprets the Next input digitally, where: value = negative or zero = unoccupied (Digital OFF) value > zero = occupied (Digital ON) value = not active (NA) = algorithm defaults to the Current event and disables the optimum start / stop function With an *Input Select of "Enumeration",* the algorithm interprets the Next input data as an enumerated value that follows the SNVT_occupancy definition: 0 = Occupied 1 = Unoccupied 2 = Bypass 3 = Standby 255 = Null A Next input value of Bypass (2), Standby (3), or Null (255), or any value outside the defined enumeration range causes the algorithm to default to the Current event and disables the optimum start / stop function.	Based upon the Input Select configuration property	In cases where both Current and Next input values are the same, either both occupied or both unoccupied, the algorithm defaults to the Current event and disables the optimum start / stop function.
Time	Time	Class: Analog - Reflects the calculated number of minutes until the next event value becomes the active event value. Time is always a positive value that decrements towards zero or the next event.	—	A Time input value that is negative or not active (NA) causes the algorithm to default to the Current event and disable the optimum start / stop function.
SetptA	Setpoint A	Class: Analog - Defines the occupied cooling setpoint target value used by the Optimum Start / Stop algorithm during both optimum start and optimum stop sequences.	—	For operation when Setpoint A is not active (NA), refer to the Setpoint Input / Output Validity Chart (Cooling), Table–5.228.
SetptB	Setpoint B	Class: Analog - Defines the occupied heating setpoint target value used by the Optimum Start / Stop algorithm during both optimum start and optimum stop sequences.	—	For operation when Setpoint B is not active (NA), refer to the Setpoint Input / Output Validity Chart (Heating), Table–5.229.

Table–5.225 OSS Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
UnocSPA	Unoccupied Setpoint A	Class: Analog - Defines the unoccupied cooling setpoint value.	—	For operation when Unoccupied Setpoint A is not active (NA), refer to the Setpoint Input / Output Validity Chart (Cooling), Table–5.228.
UnocSPB	Unoccupied Setpoint B	Class: Analog - Defines the unoccupied heating setpoint value.	—	For operation when Unoccupied Setpoint B is not active (NA), refer to the Setpoint Input / Output Validity Chart (Heating), Table–5.229.
Zone	Zone Temperature	Class: Analog - Monitors the zone temperature for OSS Modes, “Zone” and “Outside Air”. The control algorithm utilizes the zone temperature value for the analysis and calculation of new prestart and prestop periods.	—	An input of not active (NA) disables the optimum start / stop function, causing the control logic outputs and setpoint output values to follow the “Current” event and Force Occupied conditions, without optimum start / stop intervention.
OATemp	Outside Air Temperature	Class: Analog - Monitors the outside air temperature for OSS Mode, “Outside Air”. The control algorithm utilizes the outside air temperature value for calculation of base prestart and base prestop heating and cooling time periods.	—	An input of not active (NA) disables the optimum start / stop functions, causing the control logic outputs and setpoint output values to follow the “Current” event and Force Occupied conditions, without optimum start / stop intervention. The algorithm will ignore the Outside Air Temperature input when OSS Mode selection is “Zone”.

Table–5.225 OSS Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
OAHRef	Outside Air High Reference	Class: Analog - Defines the outside air temperature reference point where the Optimum Start / Stop algorithm begins to calculate optimum start / stop base time periods for cooling prestart and prestop control. To utilize the Outside Air High Reference value, OSS Mode must be set to “Outside Air”. Outside air temperatures above the Outside Air High Reference value cause the algorithm to calculate actual prestart and prestop base time periods, using the OAClFct and Coast values. Outside air temperatures below the Outside Air High Reference value and above the Outside Air Low Reference value cause the algorithm to bypass the cooling prestart sequence and utilize the full cooling prestop period, as defined by the Stop Limit. In this condition, prestart and prestop offsets remain unchanged. The Optimum Start / Stop object prevents outside air reference crossover by comparing the Outside Air High Reference and Outside Air Low Reference values. If Outside Air Low Reference is greater than Outside Air High Reference, then Outside Air Low Reference is made equal to Outside Air High Reference.	—	A not active (NA) causes the optimum start / stop function to be disabled for prestart / prestop cooling.

Table–5.225 OSS Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
OALRef	Outside Air Low Reference	Class: Analog - Defines the outside air temperature reference point where the Optimum Start / Stop algorithm begins to calculate the optimum start / stop base time periods for heating prestart and prestop control. To utilize the Outside Air Low Reference value, OSS Mode must be set to “Outside Air”. Outside air temperatures below the Outside Air Low Reference value cause the algorithm to calculate actual prestart and prestop base time periods using the OAHtFct and Coast values. Outside air temperatures above the Outside Air Low Reference value and below the Outside Air High Reference cause the algorithm to bypass the heating prestart sequence and utilize the full heating prestop period, as defined by the Stop Limit. In this condition, prestart and prestop offsets remain unchanged. The Optimum Start / Stop object prevents outside air reference crossover by comparing the Outside Air High Reference and Outside Air Low Reference values. If Outside Air Low Reference is greater than Outside Air High Reference, then Outside Air Low Reference is made equal to Outside Air High Reference.	—	A not active (NA) causes the optimum start / stop function to be disabled for prestart / prestop heating.

Table–5.226 OSS Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
OssSPCtl	OSS Setpoint Control	Class: Digital - Indicates the mode in which the Optimum Start / Stop algorithm is operating Digital ON = the Optimum Start / Stop algorithm is operating in the occupied mode. Digital OFF = the Optimum Start / Stop algorithm is operating in the unoccupied, optimum start, or optimum stop mode. This output is typically connected to the Setpoint Control object and is used to control setpoints during all occupied, unoccupied, and optimum start / stop conditions.	OFF (0.0) ON (100.0)

Table–5.226 OSS Object Output Properties. (Continued)

Abbrev.	Name	Class / Description	Valid Values
OssSPA	OSS Setpoint A	Class: Analog - Reflects the calculated Optimum Start / Stop cooling setpoint value. During the various modes, the algorithm functions as follows: *Unoccupied mode* — the algorithm holds the OSS Setpoint A at the Unoccupied Setpoint A value. *Optimum prestart sequence* — in the cooling mode, the algorithm ramps the controlling OSS Setpoint A output from the Unoccupied Setpoint A value to the occupied Setpoint A value, over the calculated prestart period. The ramp starts at the prestart time and finishes at the occupied time minus the associated lag time. In the heating mode, this non-controlling OSS Setpoint A output switches from the Unoccupied Setpoint A value to the occupied comfort target value at the prestart time. *Occupied mode* — the algorithm holds the OSS Setpoint A at the Setpoint A value. *Optimum prestop sequence* — the algorithm adjusts the OSS Setpoint A output to the calculated coast position setpoint level, for the calculated prestop period.	For operation when Setpoint A and / or Unoccupied Setpoint A is not active (NA), refer to the Setpoint Input / Output Validity Chart (Cooling), Table–5.228.
OssSPB	OSS Setpoint B	Class: Analog - Reflects the calculated Optimum Start / Stop heating setpoint value. During the various modes, the algorithm functions as follows: *Unoccupied mode* — the algorithm holds the OSS Setpoint B at the Unoccupied Setpoint B value. *Optimum prestart sequence* — in the heating mode, the algorithm ramps the controlling OSS Setpoint B output from the Unoccupied Setpoint B value to the occupied Setpoint B value, over the calculated prestart period. The ramp starts at the prestart time and finishes at the occupied time minus the associated lag time. In the cooling mode, this non-controlling OSS Setpoint B output switches from the Unoccupied Setpoint B value to the occupied comfort target value at the prestart time. *Occupied mode* — the algorithm holds the OSS Setpoint B at the Setpoint B value. *Optimum prestop sequence* — the algorithm adjusts the OSS Setpoin B output to the calculated coast position setpoint level, for the calculated prestop period.	For operation when Setpoint B and / or Unoccupied Setpoint B is not active (NA), refer to the Setpoint Input / Output Validity Chart (Heating), Table–5.229.
Occupied	Occupied	Class: Digital - A Digital ON at this output indicates that the Optimum Start / Stop algorithm is operating in the occupied mode. This indication is based upon the “Current” event status and the current Force Occupied conditions.	OFF (0.0) ON (100.0)
OssStart	OSS Start Status	Class: Digital - A Digital ON at this output indicates that the Optimum Start / Stop algorithm is performing an active optimum start sequence.	OFF (0.0) ON (100.0)
OssStop	OSS Stop Status	Class: Digital - A Digital ON at this output indicates that the Optimum Start / Stop algorithm is performing an active optimum stop sequence.	OFF (0.0) ON (100.0)

Additional Output Properties

Eight outputs in the OSS object are hidden by default. These output properties are typically used for diagnostic purposes and are not needed for normal operation. To make any number of these outputs visible, right-click on the OSS object, click **Customize**, click the **Outputs** tab, then select the desired output(s).

Table–5.227 OSS Object Additional Output Properties.

Abbrev.	Name	Class / Description	Range / Selection
ClStart	Cooling Prestart Period	Class: Analog - Reflects the cooling prestart time period (in minutes) calculated from the previous cooling optimum start sequence. When OSSMode is set to “Zone”, the value reflects the calculated cooling prestart period. When OSSMode is set to “Outside Air”, the value reflects the calculated cooling prestart period, which is a combination of the cooling base prestart period adjusted by the cooling prestart offset. In all cases, the output is updated upon successful completion of a cooling optimum start sequence.	0 to 16383 or NA
ClOffst	Cooling Prestart Offset	Class: Analog - Reflects the cooling prestart offset (in minutes) calculated from the previous cooling optimum start sequence. When OSSMode is set to “Zone”, the value is unused by the algorithm and remains held at a not active (NA). When OSSMode is set to “Outside Air”, the value reflects the calculated cooling prestart offset. This output is updated upon successful completion of a cooling optimum start sequence.	0 to 16383 or NA
ClLag	Cooling Lag Time	Class: Analog - Reflects the cooling lag time (in minutes) calculated from the previous cooling optimum start sequence. When OSSMode is set to “Zone” or “Outside Air”, the cooling lag time output is updated upon successful completion of a cooling optimum start sequence.	0 to 16383 or NA
HtStart	Heating Prestart Period	Class: Analog - Reflects the heating prestart time period (in minutes) calculated from the previous heating optimum start sequence. When OSSMode is set to “Zone”, the value reflects the calculated heating prestart period. When OSSMode is set to “Outside Air”, the value reflects the calculated heating prestart period, which is a combination of the heating base prestart period adjusted by the heating prestart offset. In all cases, the output is updated upon successful completion of a heating optimum start sequence.	0 to 16383 or NA

Table–5.227 OSS Object Additional Output Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection
HtOffst	Heating Prestart Offset	Class: Analog - Reflects the heating prestart offset (in minutes) calculated from the previous heating optimum start sequence. When OSSMode is set to “Zone”, the value is unused by the algorithm and remains held at a not active (NA). When OSSMode is set to “Outside Air”, the value reflects the calculated heating prestart offset. This output is updated upon successful completion of a heating optimum start sequence.	0 to 16383 or NA
HtLag	Heating Lag Time	Class: Analog - Reflects the heating lag time (in minutes), calculated from the previous heating optimum start sequence. When OSSMode is set to “Zone” or “Outside Air”, the heating lag time output is updated upon successful completion of a heating optimum start sequence.	0 to 16383 or NA
StpTm	Prestop Period	Class: Analog - Reflects the prestop time period (in minutes), calculated from the previous optimum stop sequence. When OSSMode is set to “Zone”, the value reflects the calculated prestop period. When OSSMode is set to “Outside Air”, the value reflects the calculated prestop period, which is a combination of the base prestop period adjusted by the prestop offset. In all cases, the output is updated upon successful completion of an optimum stop sequence.	0 to 16383 or NA
StpOffst	Prestop Offset	Class: Analog - Reflects the prestop offset (in minutes), calculated from the previous optimum stop sequence. When OSSMode is set to “Zone”, the value is unused by the algorithm and remains held at a not active (NA). When OSSMode is set to “Outside Air”, the value reflects the calculated prestop offset. This output is updated upon successful completion of an optimum stop sequence.	0 to 16383 or NA

Setpoint Validity Conditions

Table–5.228 and Table–5.229 describe the input and output setpoint interaction and validity based upon the Optimum Start / Stop operating modes.

Table–5.228 Setpoint Input / Output Validity Chart - Cooling.

Input Setpoint Validity		Output Value for Each Operating Mode (OssSPA)			
SteptA	UnocSPA	Unoccupied	Prestart	Occupied	Prestop
Valid	Valid	UnocSPA	Ramp	SteptA	Coast
Valid	NA	NA	NA	SetptA	Coast
NA	Valid	UnocSPA	UnocSPA	NA	NA
NA	NA	NA	NA	NA	NA

Table–5.229 Setpoint Input / Output Validity Chart - Heating.

Input Setpoint Validity		Output Value for Each Operating Mode (OssSPB)			
SteptA	**UnocSPA**	**Unoccupied**	**Prestart**	**Occupied**	**Prestop**
Valid	Valid	UnocSPB	Ramp	SteptB	Coast
Valid	NA	NA	NA	SetptB	Coast
NA	Valid	UnocSPB	UnocSPB	NA	NA
NA	NA	NA	NA	NA	NA

Applying the Object

The Optimum Start / Stop object is typically applied to HVAC systems such as air handlers, boilers, and other controlled devices that operate in both occupied and unoccupied modes. This object allows desired comfort levels to be achieved for both occupied and unoccupied periods while minimizing energy expenditure.

Optimum Start and Optimum Stop Functions

Both optimum start and optimum stop are designed to provide an energy efficient transition between occupied and unoccupied modes while satisfying the primary need for comfortable temperatures.

Optimum Start

This function is designed to achieve occupied setpoint comfort levels while using the least amount of energy. For optimum start, the algorithm calculates a start time, to occur before the scheduled occupied period, that will ramp the controlled media temperature from the *unoccupied* control setpoint to the desired *occupied* control setpoint. The start time is calculated for the latest possible moment that will still achieve the desired occupancy conditions by the occupied time. The ramping of setpoints provides a soft transition between unoccupied and occupied modes, which minimizes energy use.

Optimum Stop

This function is designed to maintain an allowable comfort level, with the least amount of energy usage, up to the start of the scheduled unoccupied event. For optimum stop, the algorithm calculates a stop time, to occur before the scheduled *unoccupied* period, that allows the temperature of the controlled media to “coast” from the *occupied* setpoint towards a different, predetermined setpoint level.

Optimum Start / Stop Modes

The Optimum Start / Stop object provides two user-selectable Optimum Start / Stop modes, called Zone Mode and Outside Air Mode. These provide two different methods for producing the optimum start and stop times for the controlled media. Refer to the sections, *Zone Mode* and *Outside Air Mode*, for details.

Optimum Start / Stop object event information is typically provided either internally by a control schedule (Schedule object), or externally through the use of a network variable based on the SNVT_tod_event format. In both cases, the event information is represented using three data elements that include the current event condition, the next event condition, and the time (in minutes) remaining until the next event.

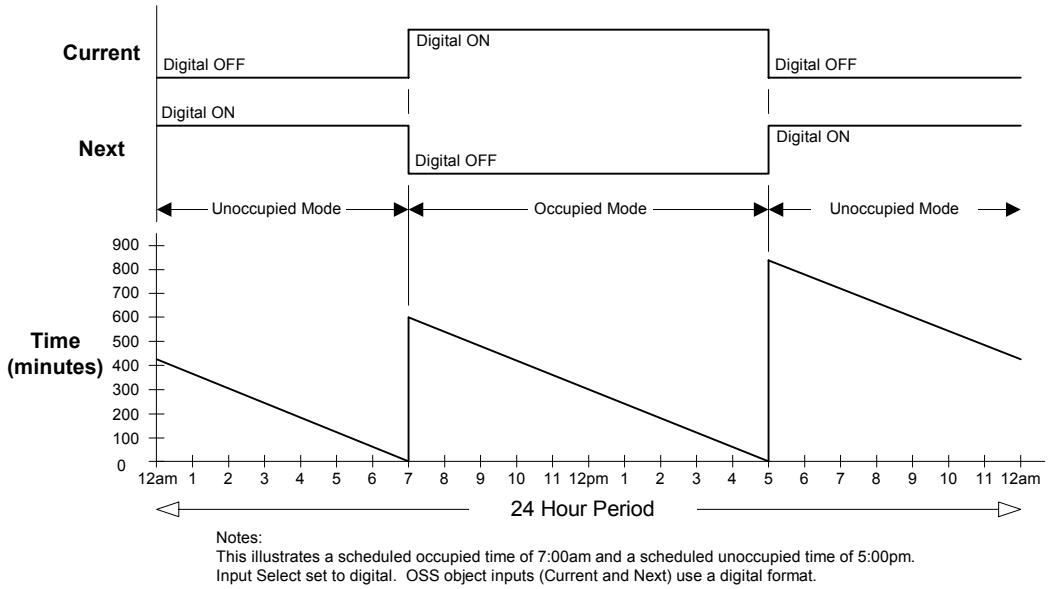
Input Select / Input Format

The Optimum Start / Stop object's Input Select configuration property defines the data format of the Current and Next inputs. The Current input provides the currently active event value. The Next input provides the next scheduled event value. An Input Select value of 0 (zero) causes the algorithm to interpret the Current and Next input data digitally. An Input Select value of 1 causes the algorithm to interpret the Current and Next input data as an enumerated value that follows the SNVT_occupancy definition. An Input Select that is not active (NA), or is out of range, causes the algorithm to default to an Input Select of 0 (zero). In both cases, the Time input reflects the calculated number of minutes until the Next event value becomes the Current (active) event value. Time is always a positive value that decrements towards zero or the next event.

Input Select (Digital)

An Input Select of "Digital" causes the algorithm to interpret the Current and Next inputs digitally, where a negative or zero value represents unoccupied (Digital OFF) and a value greater than zero represents occupied (Digital ON). A Current input value of not active (NA) causes the algorithm to default to an occupied condition, disabling the optimum start / stop function. A Next input value of not active (NA) causes the algorithm to default to the Current event and disable the optimum start / stop function. A Time input value that is negative or not active (NA) causes the algorithm to default to the Current event and disable the optimum start / stop function.

In cases where both the Current and Next values are the same, either both occupied or both unoccupied, the algorithm defaults to the Current event and disables the optimum start / stop function.



Notes:
This illustrates a scheduled occupied time of 7:00am and a scheduled unoccupied time of 5:00pm.
Input Select set to digital. OSS object inputs (Current and Next) use a digital format.

Figure–5.103 Event Control Signals (InSel = Digital).

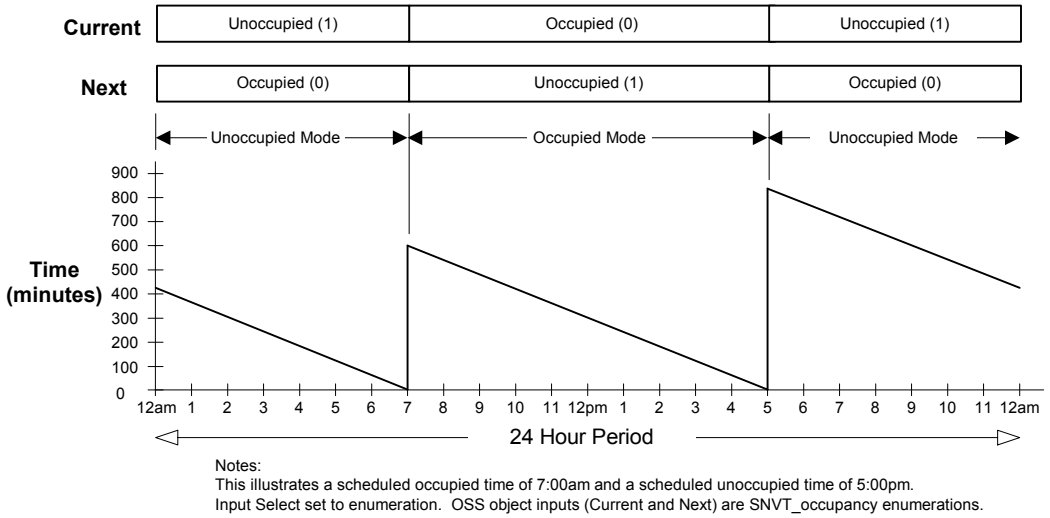
Input Select (Enumeration)

An Input Select of “Enumeration” causes the algorithm to interpret the Current and Next input data as an enumerated value that follows the SNVT_occupancy definition. SNVT_occupancy is defined by Echelon’s SNVT Master List and Programmers Guide, as follows:

0 = Occupied; 1 = Unoccupied; 2 = Bypass; 3 = Standby; and 255 = Null

Optimum start / stop is specifically designed for occupied / unoccupied control by a central system. Use of bypass and standby is typically localized to a particular zone and should not be included as part of the optimum start / stop strategy. Therefore, a Current input value of Bypass (2), Standby (3), Null (255), or any value out of the defined enumeration range causes the algorithm to default to an occupied condition, disabling the optimum start / stop function. A Next input value of Bypass (2), Standby (3), Null (255), or any value out of the defined enumeration range causes the algorithm to default to the Current event and disable the optimum start / stop function. A Time input value that is negative or not active (NA) causes the algorithm to default to the Current event and disable the optimum start / stop function.

In cases where both Current and Next values are the same, either both occupied or both unoccupied, the algorithm defaults to the Current event and disables the optimum start / stop function.



Figure–5.104 Event Control Signals (InSel = Enumeration).

Zone Mode

Selection of “Zone” for OssMode utilizes zone temperature to perform the Optimum Start / Stop function. During optimum start, the algorithm ramps the active setpoint from the unoccupied setpoint value to the occupied setpoint value and verifies the zone temperature at the scheduled occupancy time. At the end of each optimum start period, the algorithm analyzes the effectiveness of the temperature ramp-up and adjusts the start time for the next optimum start period, to improve its efficiency. During optimum stop, the algorithm adjusts the occupied setpoints to coast positions and monitors the resulting changes in the zone temperature. At the end of the optimum stop period, the algorithm analyzes its performance and recalculates the stop time for the next optimum stop period, to achieve greater efficiency.

The controller provides a means for retaining all appropriate prestart, prestop, and lag time calculated values, even through a power reset cycle, in order to maintain a reference that reflects the current building conditions.

Controllers with OSS functionality (MN 800) will provide backup. However, a controller which is allowed to lose its backup values will cause the algorithm to default the values to the following conditions:

- The calculated heating and cooling prestart values, on initial prestart sequences, will be set to one half the value of the assigned heating or cooling start limit.
- The calculated prestop value, on initial prestop sequences, will default to one half the assigned stop limit.
- Cooling and heating lag time adjustments will default to zero during power reset conditions.

As a general rule, whenever the HVAC equipment has been disabled or overridden to an inactive condition, the Optimum Start / Stop object should always be disabled (OssEnb set to Digital OFF). This prevents the Optimum Start / Stop algorithm from implementing improper prestart and prestop period adjustments when the HVAC equipment cannot respond to Optimum Start / Stop setpoint changes.

Optimum Start (Prestart) Sequence

The Optimum Start / Stop algorithm monitors the event information represented by the Current, Next, and Time inputs. For prestart conditions, the Current input is unoccupied, the Next input is occupied, and the Time input decrements towards zero, reflecting the amount of minutes remaining until the scheduled occupied event.

The algorithm also monitors the current zone temperature conditions and determines the zone requirements (i.e. heat prestart or cool prestart). A zone temperature that is above the calculated cooling comfort target temperature causes the algorithm to use the cooling prestart period for optimum start control. A zone temperature that is below the calculated heating comfort target temperature causes the algorithm to use the heating prestart period for optimum start control. A zone temperature that remains within the calculated comfort target band (the area between the heating and cooling comfort targets) causes the algorithm to bypass the optimum start sequence and hold the previously calculated heating and cooling prestart periods, for use during the next active prestart sequence. A zone temperature that is both above and below the calculated cooling and heating comfort targets (i.e. it crosses the cooling and heating setpoints) causes the algorithm to use the cooling prestart conditions for optimum start control.

Optimum start occurs when the event information “Time” value crosses the calculated prestart time value, which is based upon the performance of the previous optimum start sequence. Once initiated, the optimum start sequence proceeds to achieve the occupied setpoint conditions, regardless of changing zone temperatures.

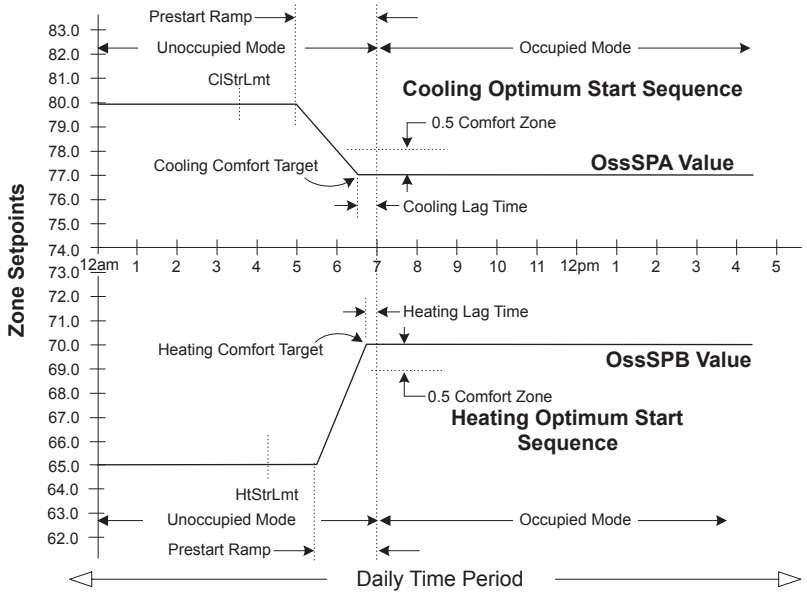
The Cooling and Heating Comfort Targets are calculated as follows:

- Cooling Comfort Target = SetptA + (Comfort ÷ 2)
- Heating Comfort Target = SetptB – (Comfort ÷ 2)

During the Optimum Start sequence, the algorithm ramps the controlling OssSPA (cooling) or OssSPB (heating) setpoint output, from the unoccupied to the occupied value. The algorithm reaches the occupied setpoint at an occupied time that is adjusted to compensate for the system temperature lag that was measured during the previous prestart sequence. These Heating and Cooling Lag Times modify the associated setpoint slope (rate) so as to achieve occupied setpoint temperatures at the required occupied event time.

The OssSPA output typically represents the cooling setpoint value and progresses from the UnocSPA value to the SetptA value. The OssSPB value typically represents the heating setpoint value, and progresses from the UnocSPB value to the SetptB value. Updates of the calculated setpoint values occur at a resolution defined by the Time input (minutes). The prestart sequence ramps the appropriate heating or cooling setpoint towards the associated occupied setpoint value. Simultaneously, the non-controlling setpoint is adjusted to its associated comfort target value for the duration of the prestart sequence. The logic outputs indicate the optimum start sequence by setting the OssStart output to Digital ON while the OssSPCtl, Occupied, and OssStop outputs are held at the Digital OFF state.

During the prestart sequence, the Optimum Start / Stop algorithm continuously monitors the zone temperature lag, by comparing the present zone temperature to the appropriate ramping setpoint value. This allows the algorithm to generate an average temperature lag for the current heating or cooling prestart sequence. In turn, this average lag is used to make any necessary adjustments to the slope (rate) of the next heating or cooling prestart setpoint.



Notes:
This illustrates an optimum start sequence that is scheduled to reach occupied setpoints at 7:00am. Heating Start Limit is set to 140 minutes and Cooling Start Limit is set to 210 minutes. The prestart sequence ramps the appropriate heating or cooling setpoint towards the associated occupied setpoint value. The non-controlling setpoint is adjusted to its associated comfort target value for the duration of the prestart sequence. At occupied time, both setpoints are set to the occupied setpoint values.

Figure–5.105 Zone Mode — Example Optimum Start Sequence.

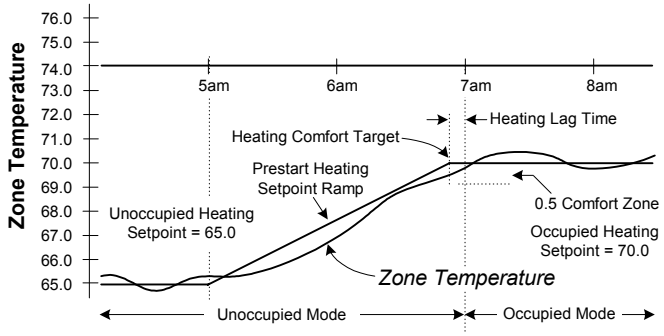
When the occupied time is reached, the Optimum Start / Stop algorithm sets both setpoints to the occupied values and reviews the zone temperature performance. The algorithm adjusts the controlling Prestart Period and associated Heating or Cooling Lag Time as required. These adjustments to the controlling Prestart Period are made by comparing the actual zone temperature to the appropriate Cooling or Heating Comfort Target.

When an optimum start sequence allows the actual zone temperature to reach the appropriate heating or cooling comfort target, it means that the previously calculated prestart period was sufficient for the HVAC equipment to meet the occupied temperature comfort requirements. This success allows the Optimum Start / Stop algorithm to shorten the prestart period, so as to conserve energy during the next optimum start sequence.

When an optimum start sequence does not result in achieving the appropriate heating or cooling comfort target, it means that the previously calculated prestart period was insufficient for the HVAC equipment to meet the comfort target requirements. In this case, the algorithm lengthens the prestart period for the next optimum start sequence, in an effort to meet the occupied setpoint comfort levels.

Any condition that disables or overrides an optimum start sequence in process (i.e. OSS Enable or Force Occupied) causes the algorithm to bypass the calculation of a new Prestart Period and Lag Time for that particular heating or cooling prestart sequence. Regardless of the disruption, the algorithm internally maintains the prestart ramp sequence, so that in the event the object disable or Force Occupied conditions are removed during the prestart period, the outputs are returned to their respective “in process” values.

The Prestart Control (Heating) illustration shows the effects of the control algorithm on the zone temperature during a heating prestart sequence. At the calculated optimum heating prestart time of 5 a.m., the heating setpoint is ramped towards the occupied heating setpoint value and the zone temperature is allowed to progress toward this control point. At the occupied time of 7 a.m., the Optimum Start / Stop algorithm performs the prestart calculations and retains the heating prestart value for the next heating prestart event.



Figure–5.106 Zone Mode — Example Prestart Control (Heating).

Optimum Stop (Prestop) Sequence

The Optimum Start / Stop algorithm monitors the event information represented by the Current, Next, and Time inputs. For prestop conditions, the Current input is occupied, the Next input is unoccupied, and the Time input decrements towards zero (reflecting the number of minutes remaining until the scheduled unoccupied event). Optimum stop occurs when the event information “Time” value crosses the calculated prestop time value, which is based upon the performance of the previous optimum stop sequence.

The optimum stop sequence adjusts the OssSPA and OssSPB setpoint outputs to new levels that widen the overall occupancy comfort range. The calculated occupied “coast” setpoint values will be maintained over the duration of the calculated prestop period. The OssSPA output typically represents the cooling setpoint value and is adjusted to a new level, based upon SetptA and the value assigned to Comfort Zone. The OssSPB output typically represents the heating setpoint value and is adjusted to a new level, based upon SetptB and the value assigned to Comfort Zone. The logic outputs indicate the optimum stop sequence by setting the OssStop and Occupied outputs to Digital ON while the OssSPCtl and OssStart outputs are held at the Digital OFF state.

The “coast” cooling and heating setpoint values are calculated as follows:

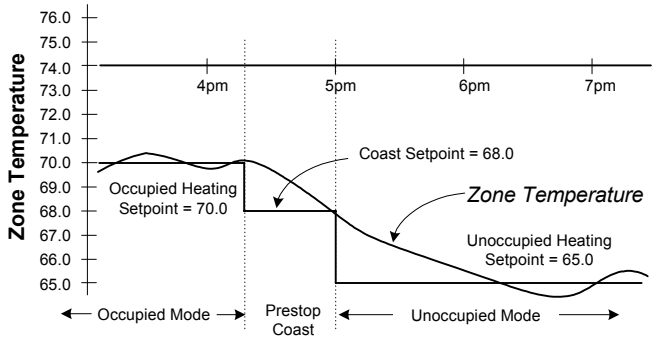
- OssSPA (Cooling) = SetptA + Comfort Zone
- OssSPB (Heating) = SetptB – Comfort Zone

During the prestop sequence, the Optimum Start / Stop algorithm compares the actual zone temperature with the appropriate “coast” setpoint value, and makes adjustments as necessary. When a zone temperature, while coasting, remains within the “coast” setpoint range for the entire duration of the prestop period, it means that the calculated prestop period was sufficient for the HVAC equipment to successfully maintain the “coast” temperature requirements. In this case, the algorithm lengthens the prestop period for the next optimum stop sequence, to conserve energy.

When zone temperatures exceed the “coast” setpoint range during the prestop sequence, it means that the calculated prestop period required the HVAC equipment to operate in order to maintain the “coast” temperature requirements. In this case, the algorithm shortens the prestop period for the next optimum stop sequence, in an effort to maintain the “coast” setpoint comfort levels.

Any condition that disables or overrides an optimum stop sequence in process (i.e. OSS Enable or Force Occupied) causes the algorithm to bypass calculation of a new Prestop Period for that particular prestop sequence. Regardless of the disruption, the algorithm internally maintains the prestop ramp sequence, so that in the event the object disable or Force Occupied conditions are removed during the prestop period, the outputs are returned to their respective “in process” values.

The Prestop Control (Heating) illustration shows the effects of the control algorithm on the zone temperature during a prestop sequence. At the calculated optimum prestop time of 4:15 p.m., the setpoints are shifted to the calculated occupied “coast” setpoint values and the zone temperature is allowed to coast toward these control points. At the unoccupied time of 5 p.m., the Optimum Start / Stop algorithm performs the prestop calculations and retains the prestop value for the next prestop event.



Figure–5.107 Zone Mode — Example Prestop Control (Heating).

Selection of Zone Heating and Zone Cooling Factors

When the required occupied temperatures are not met at the desired occupied time, correction factors are applied to the prestart time calculations. These pre-assigned factors are represented by the ZnHtFct and ZnClFct values. The ZnHtFct and ZnClFct values should be selected with consideration for both the dynamics of the building under various outside air temperature conditions, and the heating and cooling capacity of the HVAC system.

The ZnHtFct and ZnClFct factors, utilized for the optimum start sequence, should be selected as follows:

1. Determine the number of minutes required to increase the zone temperature one degree under the worst case heating condition (100% heating capacity). Conversely, determine the number of minutes required to decrease the zone temperature one degree under the worst case cooling condition (100% cooling capacity).

   For the example discussed on page 385, the zone increases one degree in a 10 minute time period during heating control (100% heating capacity when the outside air is at 0 °F) and decreases one degree in a 20 minute period during cooling control (100% cooling capacity when the outside air is at 85 °F).

2. Calculate the ZnHtFct and ZnClFct factors based upon a 50% usage of HVAC equipment capacity during the optimum start sequence. Calculate the ZnHtFct and ZnClFct factors by adjusting the previously determined heating and cooling time periods, as follows:

   Calculate ZnHtFct:

   ZnHtFct = Worst Case Heating Time ÷ 50%
   ZnHtFct = 10 Minutes ÷ 0.50
   ZnHtFct = 20

   Calculate ZnClFct:

   ZnClFct = Worst Case Heat ÷ 50%
   ZnClFct = 20 Minutes ÷ 0.50
   ZnClFct = 40

3. Determine the maximum prestart time limits (Heating Start Limit and Cooling Start Limit) for the optimum start sequence. These start limits are set to values based upon both the ZnHtFct / ZnClFct factors at 50% HVAC equipment capacity and the number of degrees required by the unoccupied to occupied setpoint ramps.

   Heating

   Heating Start Limit = (SetptB – UnocSPB) x ZnHtFct
   Heating Start Limit = (70 °F – 65 °F) x 20
   Heating Start Limit = 100 minutes

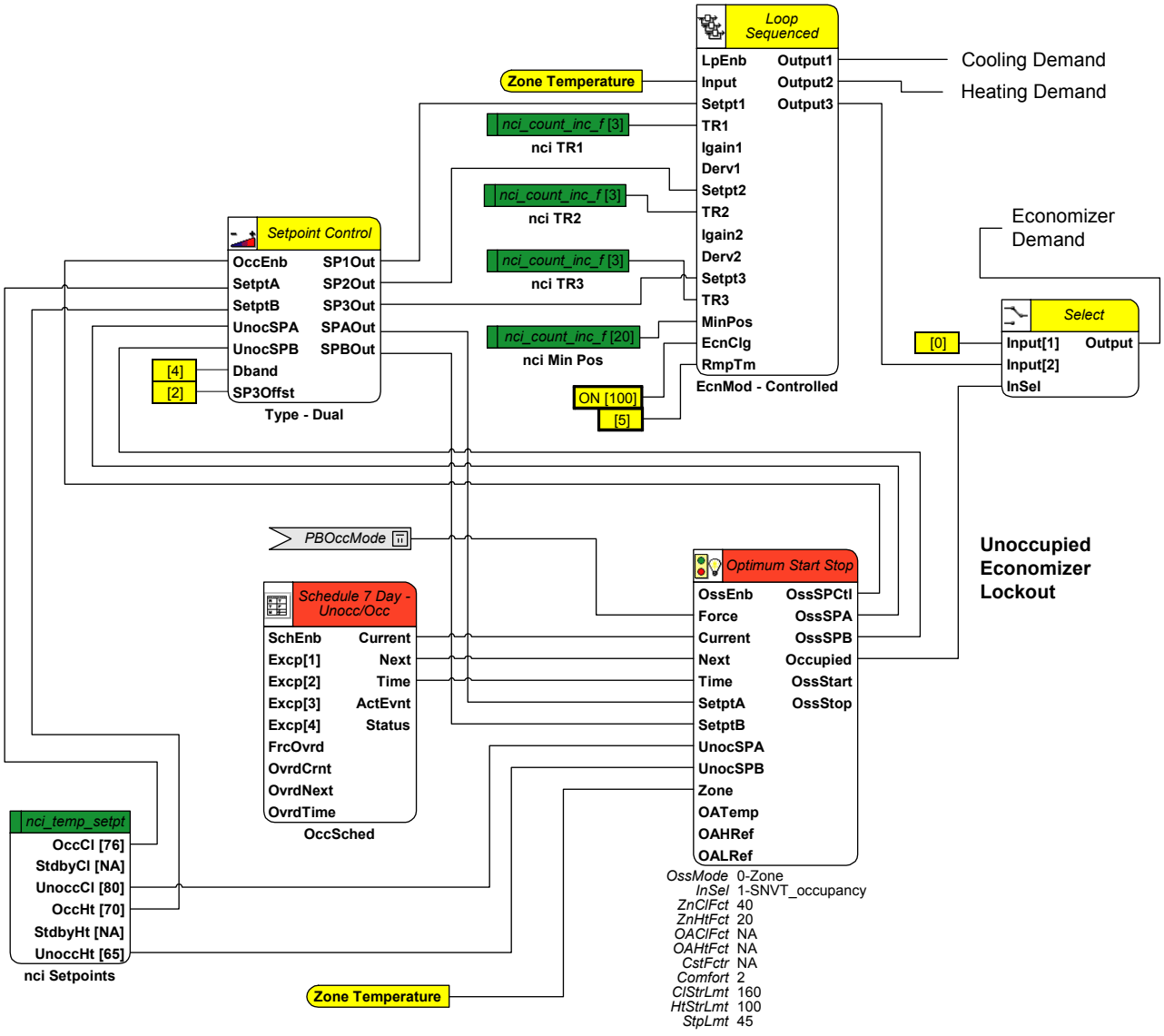
   Cooling

   Cooling Start Limit = (UnocSPA – SetptA) x ZnClFct
   Cooling Start Limit = (80 °F – 76 °F) x 40
   Cooling Start Limit = 160 minutes

The OAHtFct, OAClFct, and Coast factors are unused in the optimum stop sequence when “Zone” is selected for OssMode.

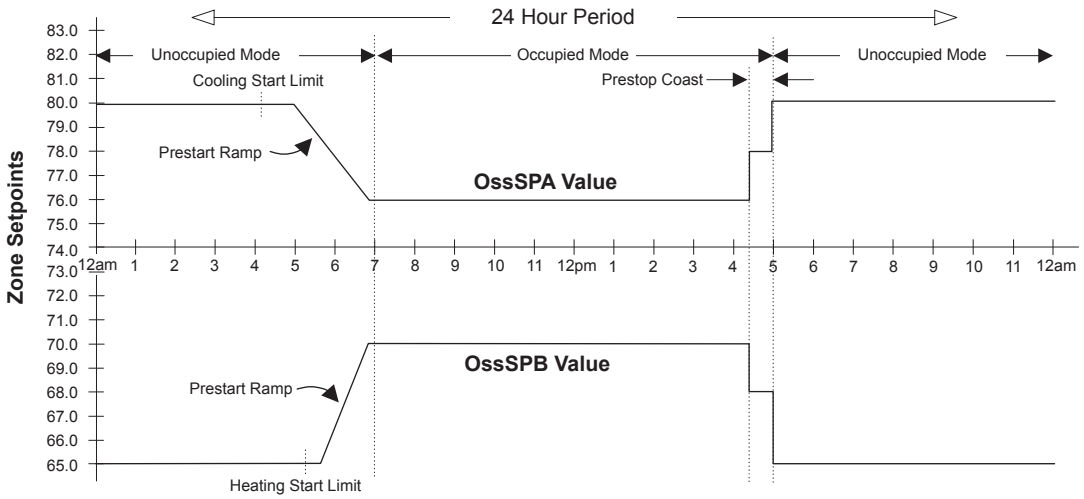
Zone Mode Example

The following example illustrates a typical Optimum Start / Stop application in which OssMode is set to Zone. This Optimum Start / Stop object supplies the heating, cooling, and economizer setpoints to a sequenced control loop. In this example, the control schedule resides in the Schedule 7 Day object which provides event information for the Optimum Start / Stop object. The event information is represented by three data elements that include the current event condition, the next event condition, and the time (in minutes) until the next event.



Figure–5.108 OSS Object — Example Zone Mode Application.

The following Optimum Start Stop Sequence diagram Figure-5.109 shows the Optimum Start / Stop sequence during a typical 24-hour event period. The Optimum Start / Stop object monitors the Time input value, which indicates the number of minutes remaining until the next occupied or unoccupied event. The algorithm determines whether prestart heat or prestart cool is needed, based upon the zone temperature control conditions. The prestart sequence is initiated when the Time input value crosses the prestart period, which was calculated based on the performance of the previous optimum start (heat or cool) sequence. The prestart sequence ramps the appropriate heating or cooling setpoint towards the associated occupied setpoint value. Simultaneously, the non-controlling setpoint value is adjusted to its associated comfort target value for the duration of the prestart sequence.



Notes:
This illustrates a scheduled occupied time of 7:00am and a scheduled unoccupied time of 5:00pm. Heating Start Limit is 100 minutes, Cooling Start Limit is 160 minutes, and the Stop Limit is 45 minutes. The setpoint ramp / prestart calculation is determined by the controlling condition (i.e. heat or cool). The non-controlling setpoint value is adjusted to its associated comfort target value for the duration of the prestart sequence. At occupied time, both setpoints are set to occupied setpoint values.

Figure–5.109 OSS Object — Example Zone Mode Sequence Diagram.

At *occupied* time, the setpoint reaches its desired occupied value and the Optimum Start / Stop algorithm performs the prestart (heating or cooling) calculations, from which corrections are stored for the next optimum start sequence. In this example, the economizer control is released and the occupied control sequence is performed until the Time input value crosses the calculated prestop value. At the optimum prestop time, the setpoints are shifted to the calculated occupied “coast” setpoint values and the zone temperature is allowed to coast toward these control points. At *unoccupied* time, the Optimum Start / Stop algorithm performs the prestop calculations, from which corrections are stored for the next optimum stop sequence. The outside air damper is closed and the unoccupied control sequence is performed until the Time input value crosses the calculated prestart value.

Outside Air Mode

Selection of “Outside Air” for OssMode utilizes both the zone and outside air temperatures to perform the Optimum Start / Stop function. In this mode, the outside air temperature sensor becomes the primary controlling factor for the calculation of new base prestart and prestop time periods. The constant monitoring of outside air temperature results in more precise prestart and prestop control by allowing immediate adjustments to be made, based on changing outside air conditions. Following each prestart and prestop period, the system’s performance is evaluated and the prestart or prestop time is recalculated as necessary, to further improve the next period’s performance.

The controller provides a means for retaining all appropriate calculated values for zone prestart, zone prestop, and lag time, through a power reset cycle so as to maintain a reference that reflects the current building conditions.

A loss of backed up values through a power reset cycle will cause the algorithm to default the values to the following conditions:

- The calculated heating and cooling zone prestart offset values will be reset to zero on *initial* prestart sequences.
- The calculated zone prestop offset value will be reset to zero on *initial* prestop sequences.
- Cooling and heating lag time adjustments will default to zero during power reset conditions.

Whenever the HVAC equipment has been disabled or overridden to an inactive condition, a general rule is to disable the Optimum Start / Stop object (OssEnb set to Digital OFF). This prevents the Optimum Start / Stop algorithm from implementing improper prestart and prestop period adjustments while the HVAC equipment is unable to respond to Optimum Start / Stop setpoint changes.

During optimum start, the algorithm monitors the outside air temperature and initiates the optimum start function at a calculated time. This time is based upon the assigned outside air heating and cooling factors, as well as a time offset that represents the zone’s optimum start performance from the previous start sequence. An outside air temperature that is above the outside air high reference value (OAHRef) causes the algorithm to use the calculated cooling prestart period for optimum start control. An outside air temperature that is below the outside air low reference value (OALRef) causes the algorithm to use the calculated heating prestart period for optimum start control. The optimum start control ramps the controlling setpoint from unoccupied to occupied over the calculated heating or cooling prestart period.

During optimum stop, the algorithm monitors the outside air temperature and, at a calculated optimum prestop time, adjusts the occupied setpoints to coast positions, to initiate the optimum stop function. The optimum prestop time calculation is based upon the assigned coast factor and a time offset that represents the zone’s optimum stop function performance from the previous stop sequence.

In all cases, an outside air temperature that remains within a temperature range as defined by the assigned outside air references (OAHRef and OALRef) causes the algorithm to bypass the heating or cooling prestart sequence and perform the full prestop period as defined by the assigned Stop Limit. All prestart and prestop offset values are retained and used when required (at the next active optimum start or stop sequence).

Optimum Start (Prestart) Sequence

The Optimum Start / Stop algorithm monitors the event information represented by the Current, Next, and Time inputs. For prestart conditions, the Current input is unoccupied, the Next input is occupied, and the Time input decrements towards zero, reflecting the number of minutes remaining until the scheduled occupied event.

Optimum start occurs when the event information “Time” value crosses the continuously calculated prestart time value, which reflects the present outside air condition, adjusted by a time factor that represents the previous optimum start performance. Once initiated, the optimum start sequence proceeds to achieve the occupied setpoint conditions, regardless of changing outside air temperatures.

Base prestart time values are calculated using the present outside air temperature, the outside air heating and cooling K factors, and the relationship of the present outside air temperature to the assigned outside air references (OAHRef and OALRef). The outside air heating and cooling K factors represent the amount of prestart minutes required for each degree above or below the appropriate outside air reference.

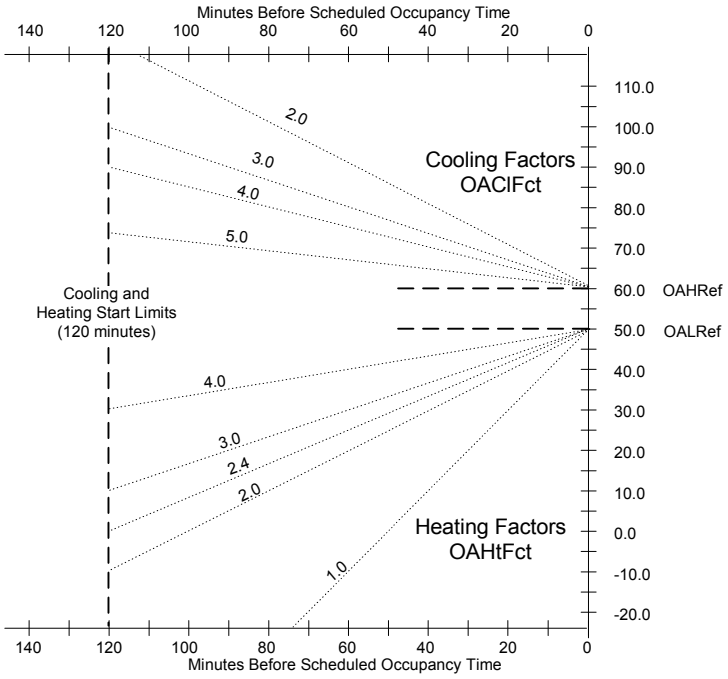
The base prestart periods for cooling and heating conditions are calculated as follows:

- Cooling Base Prestart Period = (Outside Air Temp – OAHRef) x OAClFct
- Heating Base Prestart Period = (OALRef – Outside Air Temp) x OAHtFct

Note:

- Cooling Base Prestart Periods are not allowed to be less than zero or exceed the assigned Cooling Start Limit value.
- Heating Base Prestart Periods are not allowed to be less than zero or exceed the assigned Heating Start Limit value.

The outside air heating / cooling factor relationship chart in Figure–5.110 shows how various outside air K factors affect calculated base prestart time periods. In this chart, the cooling and heating start limits have been set to 120 minutes.



Figure–5.110 Outside Air Mode — Example Heating / Cooling Factor Relationship.

The calculated heating and cooling base prestart periods are further adjusted by the addition of a heating or cooling offset time, which is calculated based on the performance of the previous optimum start sequence. The result of this calculation forms the actual prestart time period used by the algorithm to initiate the optimum start sequence.

The prestart periods for cooling and heating conditions are calculated as follows:

- Cooling Prestart Period = Cooling Base Prestart Period + Cooling Prestart Offset
- Heating Prestart Period = Heating Base Prestart Period + Heating Prestart Offset

---

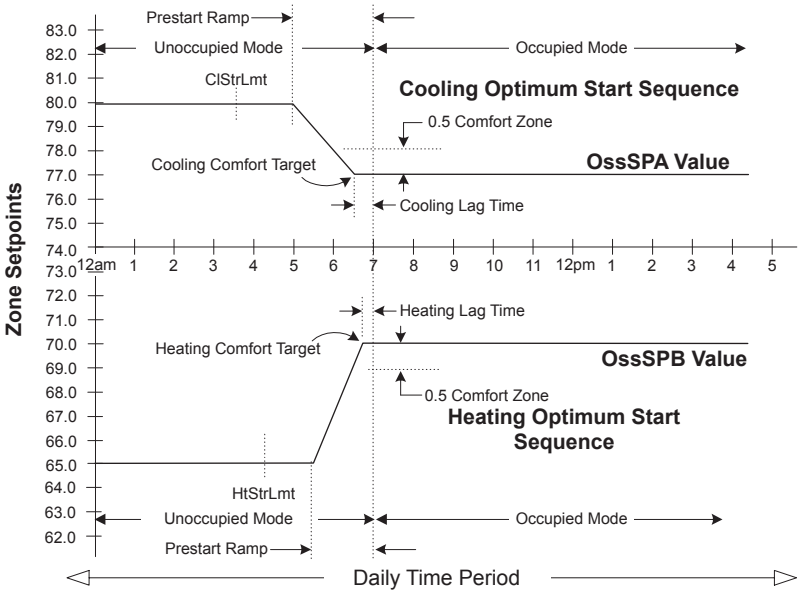
Note:

- Cooling Prestart Periods are not allowed to exceed the assigned Cooling Start Limit value.
- Heating Prestart Periods are not allowed to exceed the assigned Heating Start Limit value.

During the Optimum Start sequence, the algorithm ramps the controlling OssSPA or OssSPB setpoint output, from the unoccupied to occupied value. The algorithm reaches the occupied setpoint at an occupied time that is adjusted to compensate for the system temperature lag that was measured during the previous prestart sequence. These Heating and Cooling Lag Times modify the associated setpoint slope (rate) so as to achieve occupied setpoint temperatures at the required occupied event time.

The OssSPA output typically represents the cooling setpoint value and progresses from the UnocSPA value to the SetptA value. The OssSPB value typically represents the heating setpoint value, and progresses from the UnocSPB value to the SetptB value. Updates of the calculated setpoint values occur at a resolution defined by the Time input (minutes). The prestart sequence ramps the appropriate heating or cooling setpoint towards the associated occupied setpoint value. Simultaneously, the non-controlling setpoint value is adjusted to its associated comfort target value for the duration of the prestart sequence. The logic outputs indicate the optimum start sequence by setting the OssStart output to Digital ON while the OssSPCtl, Occupied, and OssStop outputs are held at the Digital OFF state.

During the prestart sequence, the Optimum Start / Stop algorithm continuously monitors the zone temperature lag, by comparing the present zone temperature to the appropriate ramping setpoint value. This allows the algorithm to generate an average temperature lag for the current heating or cooling prestart sequence. In turn, this average lag is used to make any adjustments to the slope (rate) of the next heating or cooling prestart setpoint.



Notes:
This illustrates an optimum start sequence that is scheduled to reach occupied setpoints at 7:00am. Heating Start Limit is set to 140 minutes and Cooling Start Limit is set to 210 minutes. The prestart sequence ramps the appropriate heating or cooling setpoint towards the associated occupied setpoint value. The non-controlling setpoint value is adjusted to its associated comfort target value for the duration of the prestart sequence. At occupied time, both setpoints are set to occupied setpoint values.

Figure–5.111 Outside Air Mode — Example Optimum Start Sequence.

When the occupied time is reached, the Optimum Start / Stop algorithm sets both setpoints to the occupied values and reviews the zone temperature performance. The algorithm adjusts the Prestart Offset and associated Heating or Cooling Lag Time as required. These adjustments to the Prestart Offset are recalculated by comparing the actual zone temperature to the appropriate Cooling or Heating Comfort Target.

The Cooling and Heating Comfort Targets are calculated as follows:

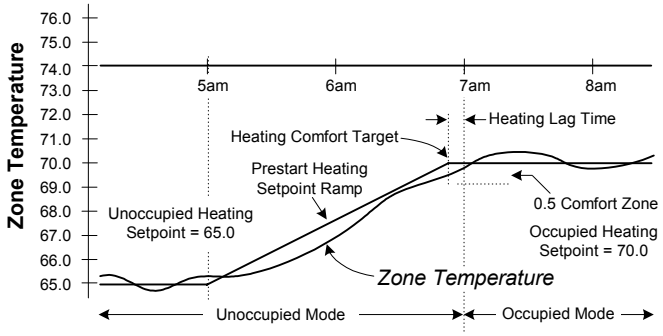
- Cooling Comfort Target = SetptA + (Comfort ÷ 2)
- Heating Comfort Target = SetptB – (Comfort ÷ 2)

When an optimum start sequence allows the actual zone temperature to reach the appropriate heating or cooling comfort target, it means that the previously calculated prestart offset time was sufficient for the HVAC equipment to meet the occupied temperature comfort requirements. This success allows the Optimum Start / Stop algorithm to shorten the prestart offset value, so as to conserve energy during the next optimum start sequence.

When an optimum start sequence does not result in achieving the appropriate heating or cooling comfort target, it means that the previously calculated prestart offset time was insufficient for the HVAC equipment to meet the comfort target requirements. In this case, the algorithm lengthens the prestart offset time value for the next optimum start sequence, in an effort to meet the occupied setpoint comfort levels.

Any condition that disables or overrides an optimum start sequence in process (i.e. OSS Enable or Force Occupied) causes the algorithm to bypass the calculation of a new Prestart Offset period and Lag Time for that particular heating or cooling prestart sequence. Regardless of the disruption, the algorithm internally maintains the prestart ramp sequence, so that in the event the object disable or Force Occupied conditions are removed during the prestart period, the outputs are returned to their respective “in process” values.

The Prestart Control (Heating) illustration Figure-5.112 shows the effects of the control algorithm on the zone temperature during a heating prestart sequence. At the calculated optimum heating prestart time of 5 a.m., the setpoint is ramped towards the occupied heating setpoint value, and the zone temperature is allowed to increase toward the control point. At the occupied time of 7 a.m., with the prestart sequence completed, the Optimum Start / Stop algorithm performs the prestart calculations and retains the heating prestart offset value for use during the next heating prestart event.



Figure–5.112 Outside Air Mode — Example Prestart Control (Heating).

Optimum Stop (Prestop) Sequence

The Optimum Start / Stop algorithm monitors the event information represented by the Current, Next, and Time inputs. For prestop conditions, the Current input is occupied, the Next input is unoccupied, and the Time input decrements towards zero (reflecting the amount of minutes remaining until the scheduled unoccupied event). Optimum stop occurs when the event information Time value crosses the continuously calculated prestop time value. The calculated prestop time value is based upon the present outside air condition, adjusted by a time factor that represents the previous optimum stop performance.

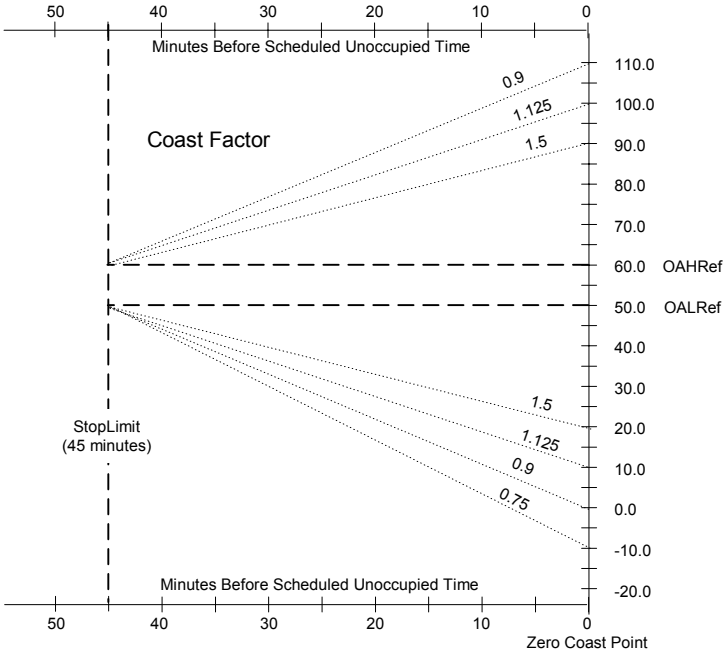
Base prestop time values are calculated using the present outside air temperature, the coast factor, and the relationship of the present outside air temperature to the assigned outside air references (OAHRef and OALRef). The coast factor represents the number of prestop minutes that are required for each degree the actual outside air temperature is above or below the appropriate outside air reference.

The Base Prestop Periods for cooling and heating conditions are calculated as follows:

- Base Prestop Period for Cooling Condition =
  Stop Limit – [(Outside Air Temp – OAHRef) x CstFctr]
- Base Prestop Period for Heating Condition =
  Stop Limit – [(OALRef – Outside Air Temp) x CstFctr]

Note: Calculated Prestop Periods are not allowed to be less than zero or exceed the assigned Stop Limit value.

Figure–5.113 shows how various coast factors affect the calculated prestop time period. In this chart, the Stop Limit has been set to 45 minutes.



Figure–5.113 Outside Air Mode — Example Coast Factor Relationship Chart for Heating and Cooling.

The calculated heating and cooling base prestop periods are further adjusted by the subtraction of a prestop offset time, which is derived from the performance of the previous optimum stop sequence. The result of this calculation forms the actual prestop time period, used by the algorithm to initiate the optimum stop sequence.

The actual prestop periods for heating and cooling conditions are calculated as follows:

Prestop Period = Base Prestop Period – Prestop Offset

Note: Prestop Periods are not allowed to be less than zero.

Once the optimum stop sequence is initiated, it will progress towards the “coast” setpoint conditions, regardless of any changes in the outside air temperature.

At optimum prestop time, the optimum stop sequence adjusts the OssSPA and OssSPB setpoint outputs to the “coast” setpoint positions that widen the overall occupancy comfort range. The calculated occupied “coast” setpoint values will be maintained over the duration of the calculated prestop period. The OssSPA output typically represents the cooling setpoint value and is adjusted to a new level based upon SetptA and the value assigned to Comfort Zone. The OssSPB output typically represents the heating setpoint value and is adjusted to a new level based upon SetptB and the value assigned to Comfort Zone. The logic outputs indicate the optimum stop sequence by setting the OssStop and Occupied outputs to Digital ON, while the OssSPCtl and OssStart outputs are held at the Digital OFF state.

The calculated “coast” cooling and heating setpoint values are calculated as follows:

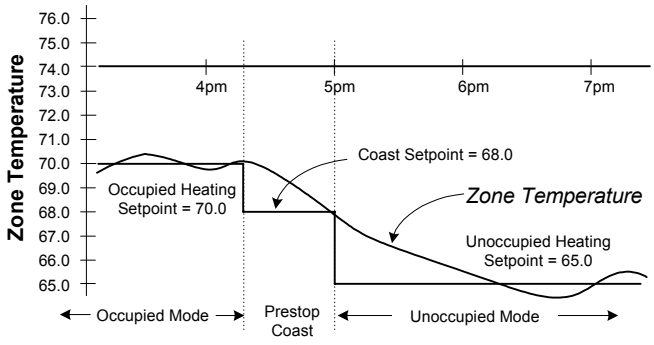
- OssSPA (Cooling) = SetptA + Comfort Zone
- OssSPB (Heating) = SetptB – Comfort Zone

During the prestop sequence, the Optimum Start / Stop algorithm compares the actual zone temperature with the appropriate “coast” setpoint value, and makes adjustments to the prestop offset as necessary. These adjustments are based upon the zone temperature performance relative to the appropriate “coast” setpoint value. When a zone temperature, while coasting, remains within the “coast” setpoint range for the entire duration of the prestop period, it means that the calculated prestop base and offset were sufficient for the HVAC equipment to successfully maintain the “coast” temperature requirements. In this case, the algorithm shortens the prestop offset, to conserve energy during the next optimum stop sequence.

When zone temperatures exceed the “coast” setpoint range during the prestop sequence, it means that the calculated prestop base and offset required the HVAC equipment to operate in order to maintain the “coast” temperature requirements. In this case, the algorithm increases the prestop offset, in an effort to maintain the “coast” setpoint comfort levels at the next optimum stop sequence.

Any condition that disables or overrides an optimum stop sequence in process (i.e. OSS Enable or Force Occupied) causes the algorithm to bypass calculation of a new Prestop Offset for that particular prestop sequence. Regardless of the disruption, the algorithm internally maintains the prestop ramp sequence, so that in the event the object disable or Force Occupied conditions are removed during the prestop period, the outputs are returned to their respective “in process” values.

The Prestop Control (Heating) illustration Figure-5.114 shows the effects of the control algorithm on the zone temperature during a prestop sequence. At the calculated optimum prestop time of 4:15 p.m., the setpoints are shifted to the calculated occupied “coast” setpoint values and the zone temperature is allowed to coast toward these control points. At the unoccupied time of 5 p.m., the Optimum Start / Stop algorithm performs the prestop calculations and retains the prestop offset value for the next prestop event.



Figure–5.114 Outside Air Mode — Example Prestop Control (Heating).

Selection of Heating, Cooling, and Coast Factors for Outside Air Mode

The following is an example of how to select heating, cooling, and coast factor values when using the Optimum Start / Stop object in the *Outside Air* mode.

The OAHtFct, OAClFct, ZnHtFct, and ZnClFct values should be selected with consideration for both the dynamics of the building under various outside temperature conditions, and the heating and cooling capacity of the HVAC system.

The heating and cooling base prestart time periods are dynamically calculated, using the OAHtFct and OAClFct values as well as the relationship of the present outside air temperature to the assigned outside air references (OAHRef and OALRef).

The heating and cooling prestart offsets are calculated using the ZnHtFct and ZnClFct values, which represent the correction factors that are applied to the prestart time calculations. These corrections are made whenever the required occupied temperatures are not properly met at the desired occupied time.

Optimum start occurs when the “Time” input value crosses the calculated prestart time value, which represents the heating or cooling base prestart time, offset by the appropriate heating or cooling offset value. Optimum start will not occur whenever the outside air temperature is within a temperature range defined by the assigned outside air references (OAHRef and OALRef). In such a case, the algorithm bypasses the heating or cooling prestart sequence and retains the prestart offsets for use during the next active optimum start sequence.

The OAHtFct and OAClFct values are used by the algorithm to calculate the base prestart period, and should be selected as follows:

1. Determine the outside air reference values for heating (OALRef) and cooling (OAHRef). These reference values define the outside air temperature points at which optimum start heating and optimum start cooling are not required. For example, a reference (OALRef) of 50 °F indicates that optimum start heating is not required whenever the current outside air temperature is above 50°F. Similarly, a reference (OAHRef) of 60 °F indicates that optimum start cooling is not required whenever the current outside air temperature is below 60 °F.
2. Determine the maximum start limit value for initiating the optimum start sequence, for both heating and cooling, and determine the outside air temperature points at which the calculated optimum start period equals the appropriate start limit time.

   Selected heating start limit assigned = 90 minutes

   Selected cooling start limit assigned = 120 minutes

   For the heating optimum start sequence, the calculated optimum start period equals the start limit value when the outside air temperature is 0 °F. For the cooling optimum start sequence, the calculated optimum start period equals the start limit value when the outside air temperature is 90 °F.
3. Calculate the heating and cooling factors from the values selected in the previous steps:

Calculate OAHtFct:

OAHtFct = HtStrLmt ÷ (OALRef) – (OA point where base prestart heat should equal Heating Start Limit)

OAHtFct = 90 minutes ÷ (50 °F – 0 °F)

OAHtFct = 1.8

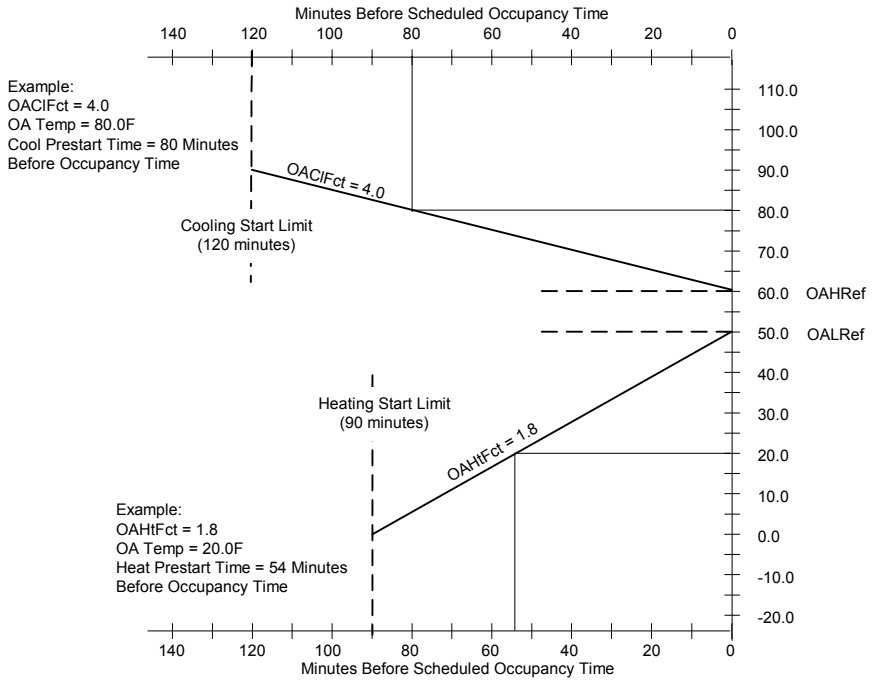
Calculate OAClFct:

OAClFct = ClStrLmt ÷ (OA point where base prestart cool should equal Cooling Start Limit – OAHRef)

OAClFct = 120 minutes ÷ (90 °F – 60 °F)

OAClFct = 4.0

Figure–5.115 illustrates the relationship of the calculated heating and cooling factors to the outside air temperature.



Figure–5.115 Outside Air Mode — Example Base Prestart Heating / Cooling.

Select the ZnHtFct and ZnClFct factors utilized for calculation of the prestart offsets, as follows:

1. Determine the number of minutes required for increasing the zone temperature one degree under the worst-case heating conditions (i.e. at 100 % heating capacity). Conversely, determine the number of minutes required for decreasing the zone temperature one degree under the worst-case cooling conditions (i.e. at 100% cooling capacity).

   For this example, the zone temperature will increase one degree in a 10 minute period during heating control (100% heating capacity when outside air is at 0 °F), and will decrease one degree in a 20 minute period during cooling control (100 % cooling capacity when outside air is at 90 °F).

2. Calculate the ZnHtFct and ZnClFct values based upon a 50 % HVAC equipment capacity:

   Calculate ZnHtFct:

   ZnHtFct = Worst Case Heat ÷ 50%

   ZnHtFct = 10 Minutes ÷ 0.50

   ZnHtFct = 20

   Calculate ZnClFct:

   ZnClFct = Worst Case Heat ÷ 50%

   ZnClFct = 20 Minutes ÷ 0.50

   ZnClFct = 40

The base prestop time value is calculated using the present outside air temperature, the Coast value, and the relationship of the present outside air temperature to the assigned outside air references (OAHRef and OALRef). The coast factor represents the number of prestop minutes required for each degree the actual outside air temperature is above or below the appropriate outside air reference. The calculated base prestop period is further adjusted by the subtraction of a prestop offset time, which is derived from the performance of the previous optimum stop sequence. The result of this calculation forms the actual prestop time period used by the algorithm to initiate the optimum stop sequence.

An outside air temperature that remains within a temperature range, as defined by the assigned outside air references (OAHRef and OALRef), causes the algorithm to perform the full prestop period as defined by the assigned Stop Limit. Optimum stop occurs when the “Time” input value crosses the prestop time period, which represents the base prestop time offset by the appropriate prestop offset value.

The Coast factor is utilized for the optimum stop sequence and is selected as follows:

1. Determine the outside air reference values for heating (OALRef) and cooling (OAHRef). These reference values define the outside air temperature points at which the optimum stop heating and optimum stop cooling functions utilize the maximum coast period allowed (Stop Limit). A reference (OALRef) of 50 °F indicates that the maximum coast period is used whenever the current outside air temperature is above 50 °F. A reference (OAHRef) of 60 °F indicates that the maximum coast period is used whenever the current outside air temperature is below 60 °F.
2. Determine the maximum stop limit value for initiating the optimum stop sequence.

   Selected stop limit assigned = 45 minutes.

3. Determine the temperature deviation from the OAHRef and OALRef references, that are needed to cause the optimum stop calculation to generate a prestop period equaling a value of zero or zero coast point. The zero coast point is the point at which prestop occurs at the unoccupied event time. Using a 50 °F temperature deviation, the zero coast points for the heating and cooling functions are determined as follows.

   Heating (zero coast point):

   Heating (zero coast point) = OALRef – zero coast temperature deviation

   Heating (zero coast point) = 50 °F – 50 °F

   Heating (zero coast point) = 0 °F

   Cooling (zero coast point):

   Cooling (zero coast point) = OAHRef + zero coast temperature deviation

   Cooling (zero coast point) = 60 °F + 50 °F

   Cooling (zero coast point) = 110 °F

4. Calculate the Coast factor from the values selected in the previous steps:

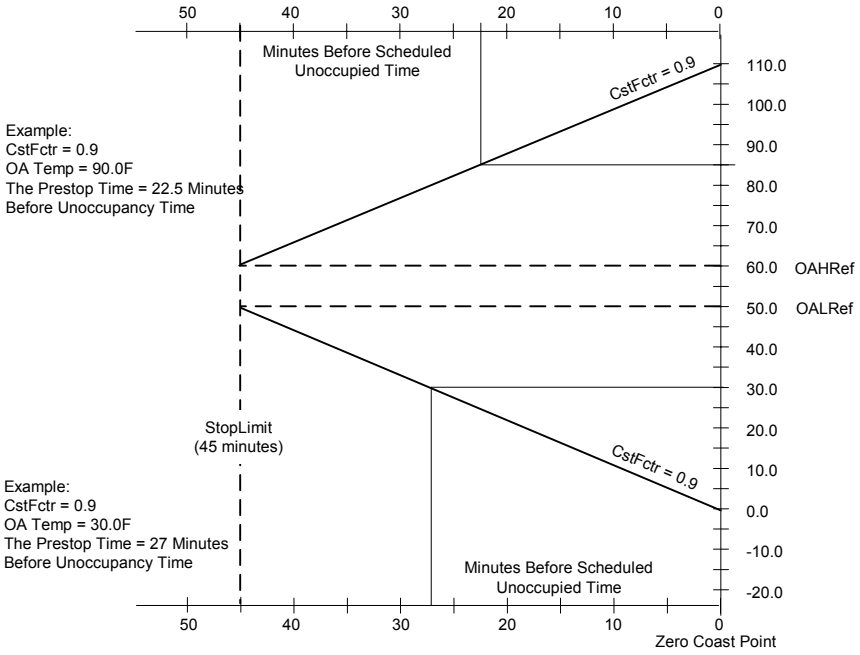
   Calculate CstFctr:

   CstFctr = Stop Limit zero coast temperature deviation

   CstFctr = 45 minutes ÷ 50 °F

   CstFctr = 0.9

Figure–5.116 illustrates the relationship of the calculated prestop coast factor to the outside air temperature.



Figure–5.116 Outside Air Mode — Example Base Prestop Period.

Utilizing the CstFctr factor, Stop Limit, and OAHRef / OALRef references, calculate the zero coast OA temperatures for heating and cooling as follows:

Heating (zero coast OA temperature):

Heating (zero coast OA temperature) = OALRef – (Stop Limit ÷ CstFctr)

Heating (zero coast OA temperature) = 50 °F – (45 minutes ÷ 0.9)

Heating (zero coast OA temperature) = 0°F

Cooling (zero coast OA temperature):

Cooling (zero coast OA temperature) = OAHRef + (Stop Limit ÷ CstFctr)

Cooling (zero coast OA temperature) = 60 °F + 50 °F

Cooling (zero coast OA temperature) = 110 °F

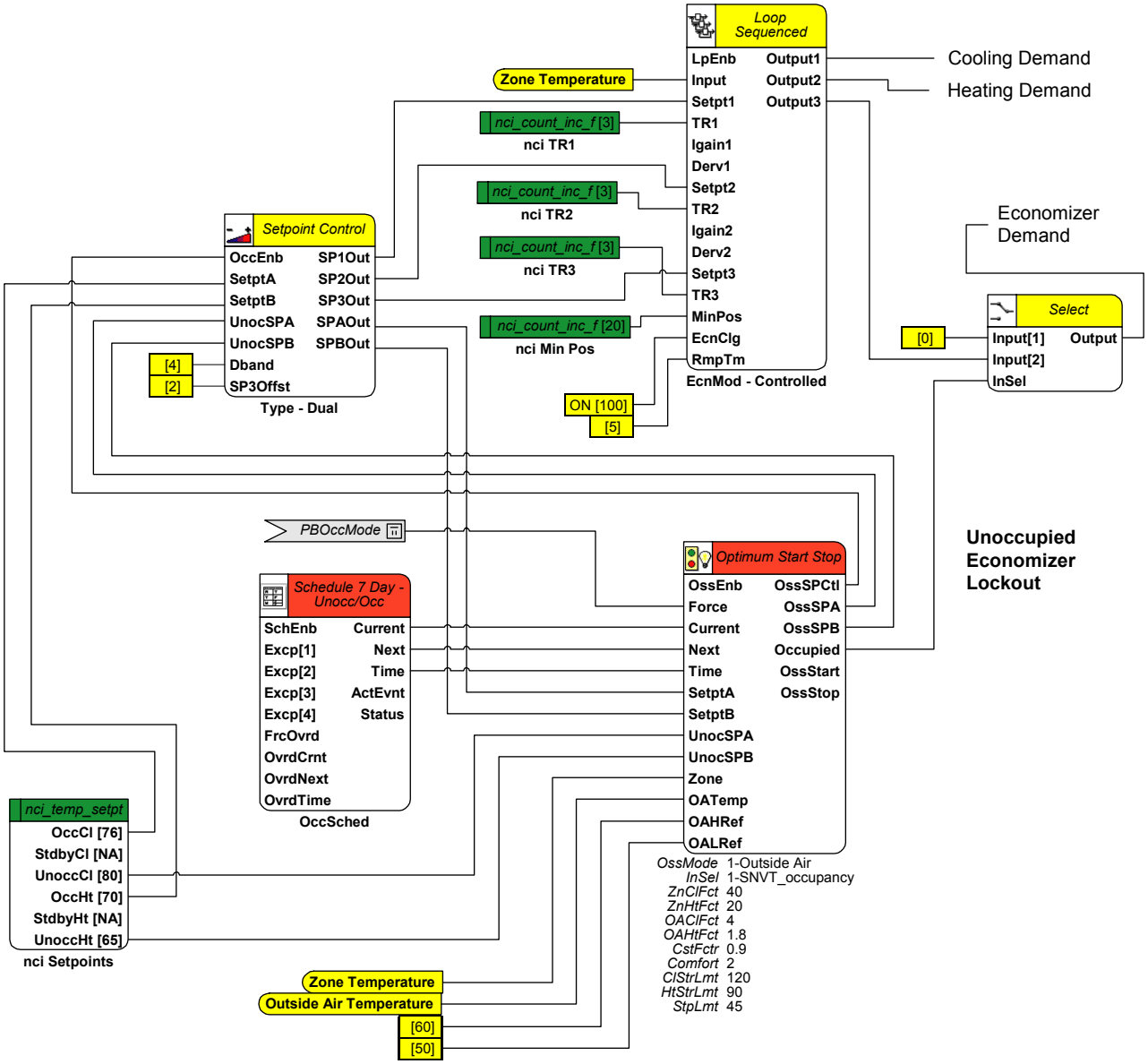
Outside Air Mode Example

Figure–5.117 illustrates an example of a typical Optimum Start / Stop (OssMode = *Outside Air*) application which supplies the heating, cooling and economizer setpoints to a sequenced control loop. For this example, the control schedule resides in the Schedule 7 Day object, which provides the event information for the Optimum Start / Stop object. The event information is represented by three data elements that include the current event condition, the next event condition, and the time (in minutes) until the next event.

Values for OAHtFct, OAClFct, ZnHtFct, ZnClFct, and CstFctr have been assigned based upon the selection process described previously. The OAHtFct, OAClFct, ZnHtFct, and ZnClFct factors should be selected to reflect the dynamics of the building, as well as the heating and cooling capacity of the HVAC system. Optimum start occurs when the “Time” input value crosses the prestart time period, which represents the heating or cooling base prestart time that is offset by the appropriate heating or cooling offset value.

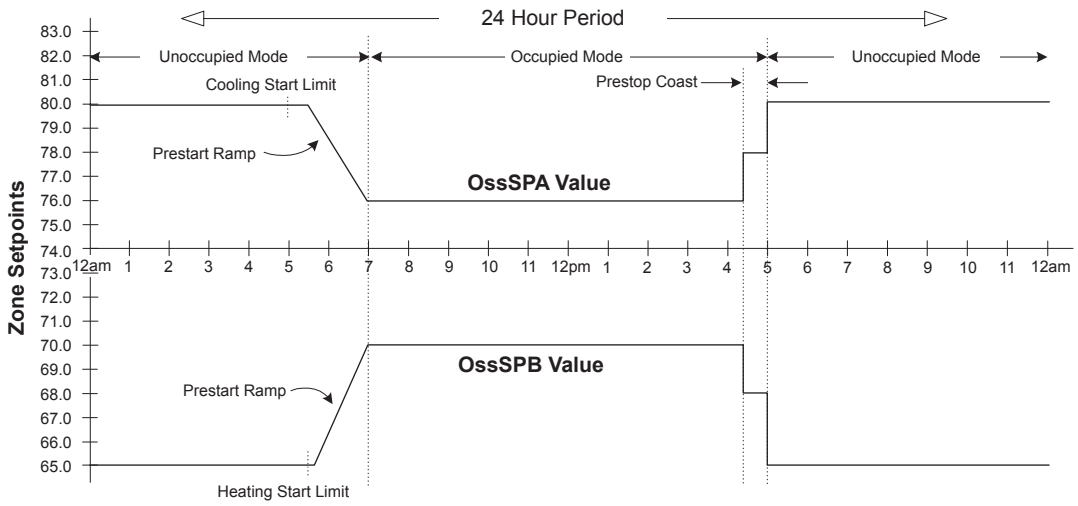
The OAHtFct value of 1.8 used in this example causes the Optimum Start / Stop algorithm to adjust the heating prestart base time period by 1.8 minutes for each degree the outside air temperature is below the OALRef reference point. Similarly, the example OAClFct value of 4.0 causes the Optimum Start / Stop algorithm to adjust the cooling prestart base time period by 4.0 minutes for each degree the outside air temperature is above the OAHRef reference point.

The CstFctr factor should be selected to reflect the dynamics of the building in varying outside air temperatures. The example CstFctr value of 0.9 causes the Optimum Start / Stop algorithm to adjust the prestop base time period by 0.9 minutes for each degree the outside air temperature is either below the OALRef reference point or above the OAHRef reference point.



Figure–5.117 OSS Object Example — Providing Setpoint Control for a Loop Sequenced Object.

The following Optimum Start Stop Sequence diagram Figure-5.118 shows the Optimum Start / Stop sequence during a typical 24 hour event period. The Optimum Start / Stop object monitors the “Time” input value, which indicates the number of minutes remaining until the next occupied or unoccupied event. Based upon the sensed outside air temperature, the algorithm determines whether prestart heat or prestart cool is required. The prestart sequence initiates when the “Time” input value crosses the calculated prestart period, which is based on the heating or cooling base prestart time, offset by the appropriate heating or cooling offset value. The prestart sequence ramps the appropriate heating or cooling setpoint towards the associated occupied setpoint value. Simultaneously, the non-controlling setpoint value is adjusted to its associated comfort target value for the duration of the prestart sequence.



Notes:
This illustrates a scheduled occupied time of 7:00am and a scheduled unoccupied time of 5:00pm. Heating Start Limit is 90 minutes, Cooling Start Limit is 120 minutes, and the Stop Limit is 45 minutes. The setpoint ramp / prestart calculation is determined by the controlling condition (i.e. heat or cool). The non-controlling setpoint value is adjusted to its associated comfort target value for the duration of the prestart sequence. At occupied time, both setpoints are set to occupied setpoint values.

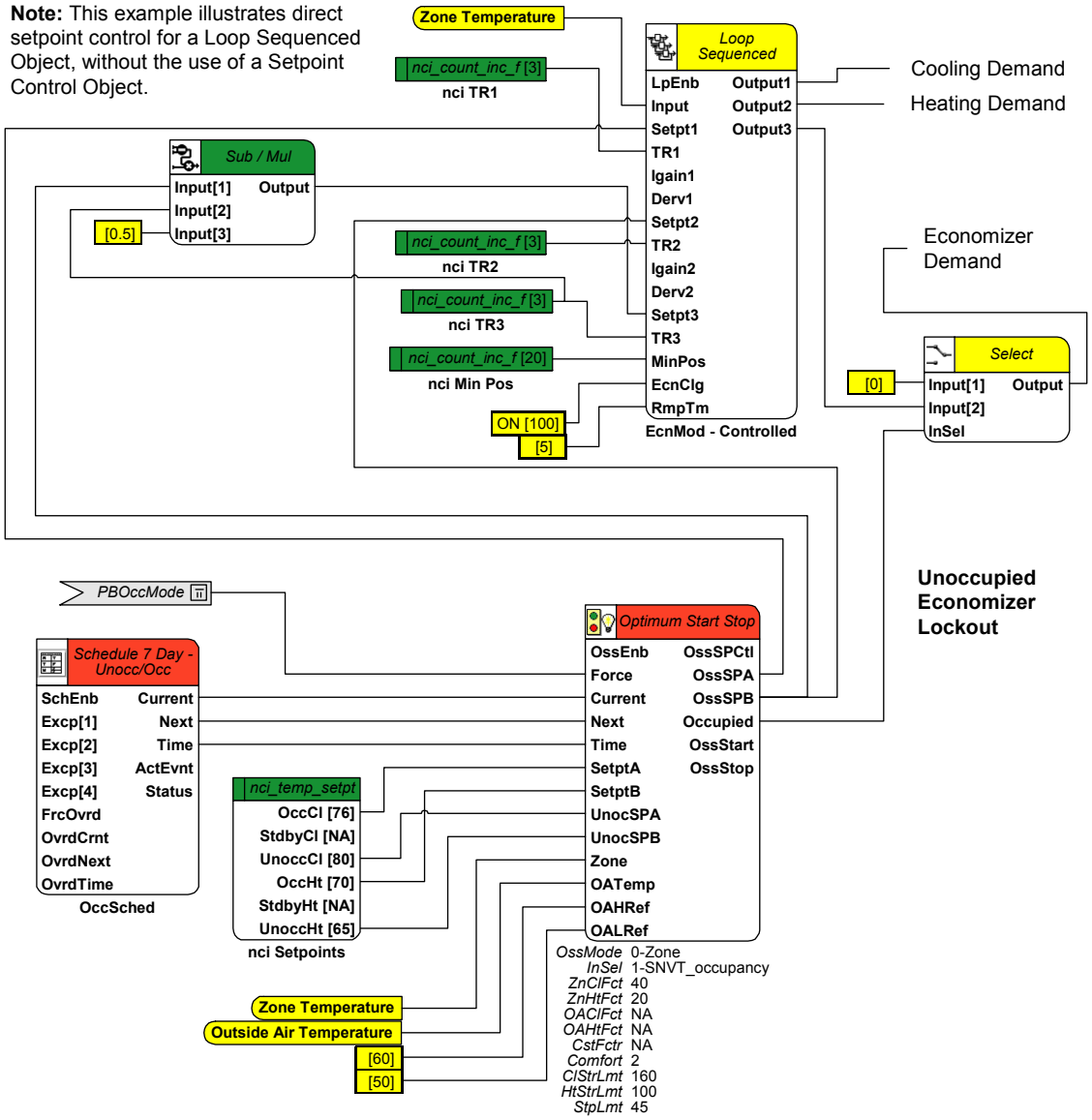
Figure–5.118 OSS Object — Example Optimum Start / Stop Sequence.

The setpoint reaches its desired occupied value at occupied time, at which point the Optimum Start / Stop algorithm performs the (heating or cooling) prestart calculations from which offsets are stored for the next optimum start sequence. In this example, the economizer control is released and the occupied control sequence is performed until the time at which the prestop sequence initiates. This occurs when the “Time” input value crosses the calculated base prestop period, offset by the prestop offset value. At the optimum prestop time, the setpoints are shifted to the calculated occupied “coast” setpoint values and the zone temperature is allowed to coast toward these control points. The algorithm monitors the coast performance and adjusts the offset value as required, for use during the next prestop sequence. Upon reaching the unoccupied time, the setpoints are set to the unoccupied setpoint values, the outside air damper is closed, and the unoccupied control sequence is performed until the “Time” input value crosses the calculated prestart value.

Application Examples

Setpoint Control without a Setpoint Control Object

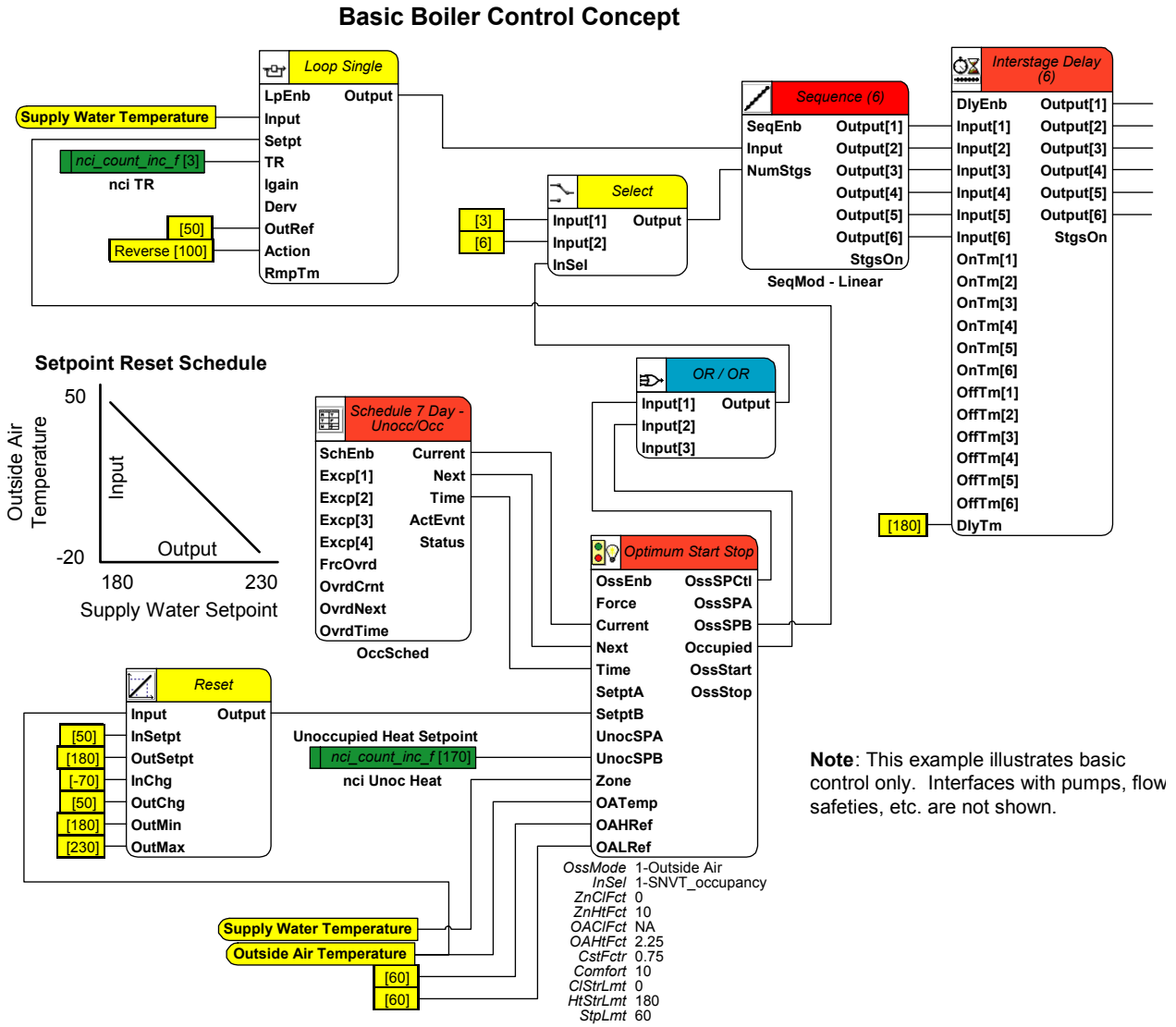
The example illustrated in Figure–5.119 presents a method of directly providing setpoint control from the Optimum Start / Stop object, without the use of a Setpoint Control object.



Figure–5.119 OSS Object Application Example — Direct Setpoint Control without a Setpoint Control Object.

Boiler Application

Figure–5.120 shows a method of applying the optimum start / stop function within a boiler application.



Figure–5.120 OSS Object Application Example — Basic Boiler Control Concept.

The unoccupied idle control setpoint for the boiler is 170 °F, and the occupied control (using setpoint reset) operates between 180 °F and 230 °F, dependent upon the outside air temperature. In this application, the design considerations for the optimum start and stop control algorithm must be based on the operational worst case conditions. Because this is a boiler application, the Cool factor, cooling start limit, and OA high reference are unused. The OAHRef is set to the OALRef value, to allow the OALRef to operate properly.

The OAHtFct value is utilized for the optimum start sequence and would be selected as follows:

1. First, determine the outside air reference value for heating (OALRef). The reference value defines the outside air temperature at which optimum start heating is not required. The reference (OALRef) of 60 °F use in this example indicates that optimum start heating is not required whenever the current outside air temperature is above 60 °F. A reference (OAHRef) of 60°F must be set to allow the OALRef to operate properly.
2. Next, determine the maximum start limit value for initiating the optimum start sequence for heating, then identify the outside air temperature at which the calculated optimum base prestart period equals the appropriate start limit time:
   a. Selected heating start limit = 180 minutes
   b. For optimum start sequence (heating), the calculated optimum base prestart period equals the start limit value when the outside air temperature is -20 °F.
3. Calculate the heating factor (OAHtFct) from the values selected in the previous steps:

   OAHtFct = HtStrLmt ÷ (OALRef, the OA point at which base prestart heat should equal Heating Start Limit)

   OAHtFct = 180 minutes ÷ (60 °F – (-20 °F))

   OAHtFct = 2.25

The ZnHtFct value reflects the dynamics of the heating capacity of the boiler system. ZnHtFct is used in the calculation of the heating prestart offset and is selected as follows

1. Determine the number of minutes required to increase the supply water temperature one degree under the worst-case heating conditions (i.e. at 100% heating capacity). In this example, the supply water temperature will increase one degree over a 5 minute time period during heating control (100% heating capacity when outside air at -20 °F).
2. Calculate the ZnHtFct value, based upon a 50% boiler equipment capacity:

   ZnHtFct = Worst Case Heat ÷ 50%

   ZnHtFct = 5 Minutes ÷ 0.50

   ZnHtFct = 10

The Coast factor is utilized for the optimum stop sequence and is selected as follows:

1. Determine the outside air low reference value (OALRef) for heating. This reference value defines the outside air temperature at which the optimum stop heating utilizes the maximum coast period allowed (Stop Limit). An OALRef value of 60 °F indicates that the maximum coast period is used whenever the outside air temperature is above 60 °F.
2. Determine the maximum stop limit value for initiating the optimum stop sequence:

   Selected stop limit = 60 minutes.

3. From the OALRef reference value, determine the temperature deviation that is needed to cause the optimum stop calculation to generate a prestop period that equals a value of zero (zero coast point). At the zero coast point, prestop occurs at the unoccupied event time, so that the time duration of the prestop sequence is zero and no “coasting” occurs. Using an 80 °F temperature deviation as an example, the zero coast point for heating is calculated as follows:

   Heating (zero coast point) = OALRef – zero coast temperature deviation

   Heating (zero coast point) = 60 °F – 80 °F

   Heating (zero coast point) = -20 °F

4. Calculate the Coast factor (CstFctr) from the values selected in the previous steps:

   CstFctr = Stop Limit ÷ zero coast temperature deviation

   CstFctr =60 minutes ÷ 80°F

   CstFctr = 0.75

Pressure Transducer

Object Usage: The Pressure Transducer object is a point-type object that supports the on-board velocity-pressure transducer on an I/A Series MicroNet VAV controller. Object outputs provide the velocity pressure value and an airflow value based upon the controller pressure units selected. The object also provides High and Low Flow Calibration inputs used to establish and modify VAV box flow parameters, during air-balancing procedures. The Pressure Transducer object typically connects to another point-type object specific to the MicroNet VAV controller, the VAV Actuator object (page 526), which modulates the integral (or external) damper actuator of the VAV controller.

Device Support: (all firmware revisions)

MNL-V1Rxx, -V2Rxx, -V3Rxx,
where xx = V1, V2, or V3

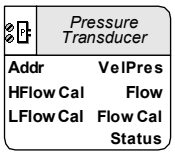
Memory Requirements: (per object)

EEPROM: 10 bytes RAM: 18 bytes

WP Tech Representation

Inputs

Physical Address
High Flow Calibration
Low Flow Calibration



Outputs

Velocity Pressure
Flow
Flow Calibration Output
Status Flags

Configuration Properties

Object Name
Object Description
Process Time

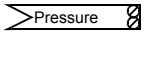
WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.230 Pressure Transducer Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.231 Pressure Transducer Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Addr	Physical Address	Class: Analog - Defines the physical hardware address assigned to the Pressure Transducer object. If not active (NA) or an invalid hardware address, the Velocity Pressure and Flow outputs are set to NA, and an error condition is indicated with an ON at the Status Flags output.	Pressure input (only)	The only valid WP Tech Hardware Input Tag is shown below. 

Table–5.231 Pressure Transducer Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
HFlowCal	High Flow Calibration	Class: Analog - Defines the equivalent input flow at 1.0 inch of WC (249 Pa) differential pressure. A not active (NA) or value of 0 (zero) or less causes a NA at outputs Velocity Pressure, Flow, and Flow Calibration, and an error condition shown with an ON Status Flags output.	0.1 to 16383	Typically this input is connected to the Profile Tag nciHighFlowCal to allow air balancing by Invensys or third-party PC programs.
LFlowCal	Low Flow Calibration	Class: Analog - Defines the equivalent input flow at 0.1 inch of WC (24.9 Pa) of differential pressure. A not active (NA) or value of 0 (zero) results in a derived low flow calibration point that is linearly based on the High Flow Calibration value.	0 to 16383	Typically this input is connected to the Profile Tag nciLowFlowCal to allow air balancing by Invensys or third-party PC programs.

Table–5.232 Pressure Transducer Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
VelPres	Velocity Pressure	Class: Analog - The actual velocity pressure value measured by the pressure transducer on the MicroNet VAV controller. The valid value has two possible ranges, depending on the Controller units selection: English or Metric (Inches WC or Pascals). A not active (NA) results from any of the following conditions: • The object is not assigned a valid physical address. • The data from the controller's pressure transducer hardware is invalid. • The High Flow Calibration is set to 0 (zero), a negative value, or NA.	0.00 to 2.00 (Inches WC or inWC) or 0 to 500 (Pascals or Pa)
Flow	Flow	Class: Analog - The calculated flow value based on the measured velocity pressure and the CFM or Liters per Second (l/s) values at the High Flow Calibration and Low Flow Calibration inputs. English = CFM, Metric = Liters per Second (l/s) A not active (NA) results from any of the following conditions: • The object is not assigned a valid physical address. • The data from the controller's pressure transducer hardware is invalid. • The High Flow Calibration is set to 0 (zero), a negative value, or NA.	0 to 16383 (Dependent on the High Flow and Low Flow Calibration values.)
FlowCal	Flow Calibration Output	Class: Analog - Directly reflects the value at the High Flow Calibration input (that defines the equivalent flow at 1.0 inch of WC or 249 pascals). This output typically connects to a VAV Actuator object's High Flow Calibration input.	0 to 16383
Status	Status Flags	Class: Analog (or Digital) - Set to a non-zero value whenever an error condition is determined by the Pressure Transducer algorithm. The output value and associated error is defined as follows: 0 - Valid setup and normal pressure operation (no error). 1 - Physical Address set to not active (NA). 2 - High Flow Calibration set to NA. 3 - Pressure sensor failure. 4 - Pressure under-range condition (pressure < 0.1 inWC). 5 - Pressure over-range condition (pressure > 2.50 inWC). 100 (ON) - pre-Rev. 3 firmware only, indicates any of the errors above.	0, 1, 2, 3, 4, or 5 where 0 = no error Rev. 3 or higher firmware required or OFF (0.0) no error and ON (100.0) error (pre-Rev. 3)

Applying the Object

A Pressure Transducer object is a point-type object required in any MicroNet VAV control application for a MNL-V1RVx, V2RVx, or V3RVx controller, all of which have an integral velocity-pressure sensor. The Pressure Transducer object provides outputs for both velocity pressure and calculated flow.

Note: The selection of controller units (English or Metric), accessed in the Hardware Wizard of WP Tech, determines whether the object uses U.S. or international (SI) units of measure. This selection affects the values produced at the Velocity Pressure and Flow outputs, as follows:

- English: Velocity Pressure - Inches of Water Column (inWC)
  Flow - Cubic Feet per Minute (CFM)
- Metric: Velocity Pressure - Pascals (Pa)
  Flow - Liters per Second (l/s)

Calibration Values

The object provides High Flow Calibration and Low Flow Calibration inputs to establish and modify VAV box flow and air balancing values. The High Flow Calibration value is mandatory, and must be set to the specified terminal box flow in CFM (l/s) that is equivalent to 1.0 inWC (249 Pa). Low Flow Calibration is optional but can further refine the flow characteristics at low velocity pressures, increasing measured flow accuracy. If used, the Low Flow Calibration value should be the specified terminal box flow in CFM (l/s) that is equivalent to 0.1 inWC (24.9 Pa).

Object Outputs

The Velocity Pressure output reflects the velocity pressure at the sensor, either in Inches WC or in Pascals. The Velocity Pressure value ranges from 0.00 to 2.00 inWC (0 to 500 Pa).

The Flow output reflects the calculated flow in CFM (or liters per second), based on the measured velocity pressure and the calibration values at the High Flow Calibration and Low Flow Calibration inputs. Accuracy of the Flow output is determined by these user-defined calibration values.

A Flow Calibration output reflects the active value at the High Flow Calibration input. The Flow Calibration output is typically connected to a VAV Actuator object's High Flow Calibration input, which helps that object determine the proper deadband region where actuator drive is not permitted.

Finally, the Pressure Transducer object provides a Status Flags output used to signal an error condition, including an under or over range pressure condition, improper object setup, or invalid sensor data.

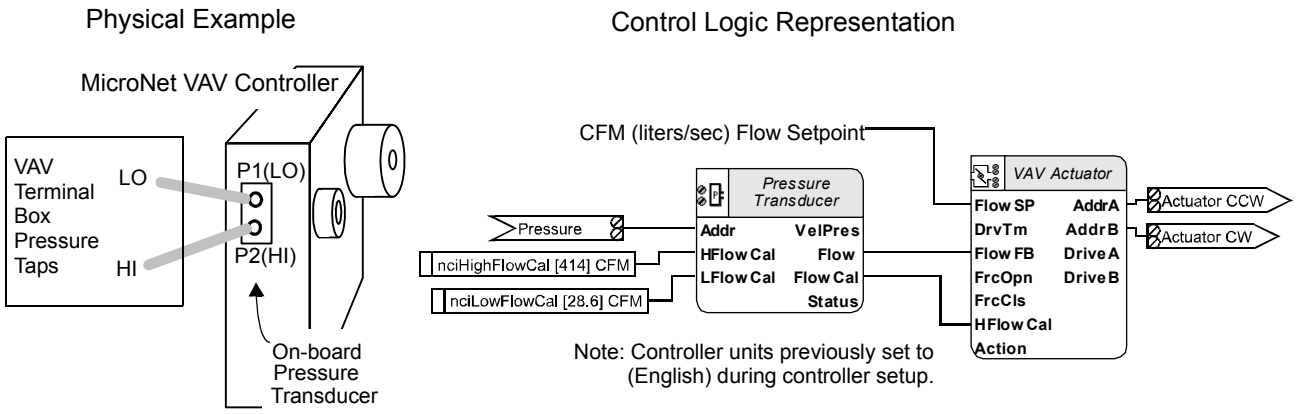
Note: In MN VAV controllers with Rev.3 or higher firmware (MNL-V1RV2, -V2RV2, -V3RV2), the Status output produces an analog value from 1 to 5 to signal a specific error type. Output in normal conditions (no error) is 0 (zero). See the Status Flags output in Table–5.232 for more details.

With MN VAV controllers with earlier firmware (MNL-V1RV1, -V2RV1, -V3RV1), this output is digital, where ON (100.0) means an error of some type and OFF (0.0) means normal conditions (no error).

Regardless of the controller firmware level, the Status Flags output can always be used digitally as a value of 0 reflects digital OFF and any value greater than zero reflects digital ON.

Example

A Pressure Transducer object is shown in Figure–5.121 below with a typical connection to a VAV Actuator object. Both of these object types are unique to the MicroNet VAV controller series, as they are special-purpose point-type objects used to support specific features of these controllers.



Figure–5.121 Example Pressure Transducer Object.

In this example, the controlled VAV terminal Box provides 414 CFM at 1.0 inches WC, and 28.6 CFM at 0.1 inches WC. The MicroNet VAV controller has an integral actuator to modulate the terminal box damper, which is controlled by the VAV Actuator object.

Calibration Overview

Proper values for the High Flow Calibration and Low Flow Calibration inputs are usually derived from running the I/A Series MicroNet Flow Balance software (WPIA-FLO), communicating online with the installed MNL-VxRVx. This procedure involves physical measurement(s) of delivered air flow (CFM or l/s) at each VAV terminal box discharge point.

Often, the initial value for nciHighFlowCal is set to a nominal value by the programmer based on the job-specific VAV box data. The value for nciLowFlowCal is left at 0, and the MicroNet Flow Balance software adjusts if required.

The Flow Balance software provides automated overrides of VAV dampers for one or more MNL-VxRVx controllers (simultaneous overrides for “groups” of controllers), allowing VAV terminal damper(s) to be positioned at settings useful for flow balancing (full open, full close, maximum flow setpoint, minimum flow setpoint, specified % position or flow rate). The Flow Balance software accesses various VAV Profile components (NCIs, NVIs, NVOs) in the MNL-VxRVx controllers.

Refer to the *I/A Series MicroNet VAV Flow Balance Manual,* F-26421, for details on running the Flow Balance software.

Priority Input (2)

Object Usage: The Priority Input (2) object automatically selects one of two valid inputs (Inputs[1] or [2]) based upon priority. Input[1] is the highest priority input, with any valid value reflected at the Output. If Input[1] is not active (NA), then any valid value at Input[2] is reflected at the Output. If both Inputs[1] and [2] are NA, the Output is NA.

The Priority Input (2) object functions like the Priority Input (4) object (page 412), except with fewer inputs. Priority Input (2) and (4) objects are commonly used in control applications to provide “fallback” logic if a higher priority input value becomes not active (NA).

Device Support: (all firmware revisions)

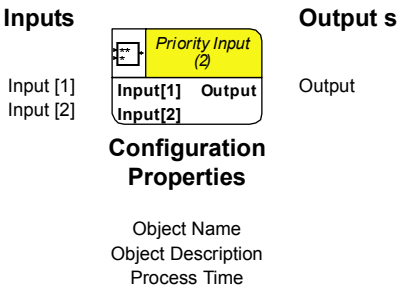
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 8 bytes
RAM: 10 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Loop and Process Control

Properties

Table–5.233 Priority Input (2) Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.234 Priority Input (2) Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The input with the highest priority. Any valid value is automatically reflected at the Output.	-163.83 to 16383	If not active (NA), Input[2] is evaluated.
Input[2]	Input [2]	Class: Analog - The input with the lowest priority. Evaluated only if Input[1] is NA, whereby any valid value is reflected at the Output.	-163.83 to 16383	If not active (NA), the Output is set to NA.

Table–5.235 Priority Input (2) Object Output Properties.

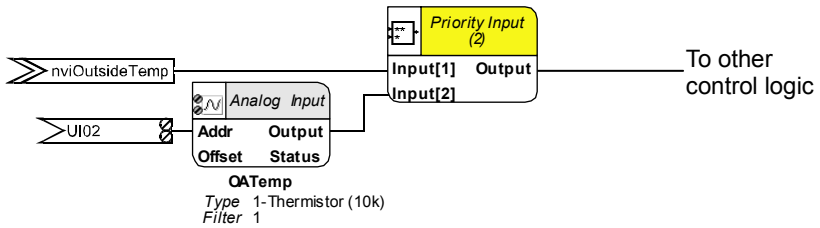
Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog -Reflects the valid value at the highest priority input (Input[1] if valid, else Input[2]). A not active (NA) output results if both inputs are NA.	-163.83 to 16383

Applying the Object

The Priority Input (2) object is typically used for “fallback” type logic within a control application. An example object is shown below Figure-5.122.

Example

Input[1] is a temperature value from a network variable input (NVI). When valid, this value is always at the object’s output for use by the application. Input[2] is a temperature value from a locally connected sensor. This value appears at the output of the Priority Input (2) object whenever the value at Input[1] (via the network) is not active (NA).



Figure–5.122 Example Priority Input (2) Object.

Priority Input (4)

Object Usage: The Priority Input (4) object automatically selects one of four valid inputs using a high (Input[1]) to low (Input[4]) priority search and passes the first valid value found. A valid value is any other than a not active (NA). If all inputs have an NA, the Output is also NA.

The Priority Input (4) object functions like the Priority Input (2) object (page 410), except with more inputs and an additional Control Level output, which indicates the active priority level (1, 2, 3, or 4). These same priority input functions are also included in these other Priority control objects:

- Analog Output Priority (page 139)
- Floating Actuator Priority (page 270)
- PWM Priority (page 427)

The Priority Input (4) and (2) objects are commonly used in applications to provide “fallback” logic if a higher priority input value becomes not active (NA).

Device Support: (all firmware revisions)

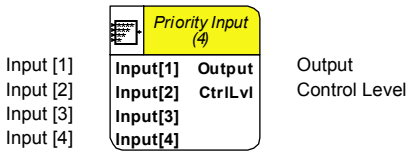
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 12 bytes
RAM: 16 bytes (standard controller)
4 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.236 Priority Input (4) Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.

Table–5.236 Priority Input (4) Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.237 Priority Input (4) Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The input with the highest priority. Any valid value is automatically reflected at the Output.	-163.83 to 16383	If not active (NA), Input[2] is evaluated.
Input[2]	Input [2]	Class: Analog - The input with the second highest priority. Evaluated only if Input[1] is NA, whereby any valid value is reflected at the Output.	-163.83 to 16383	If not active (NA), Input[3] is evaluated.
Input[3]	Input [3]	Class: Analog - The input with the third highest priority. Evaluated only if Inputs[1] and [2] are NA, whereby any valid value is reflected at the Output.	-163.83 to 16383	If not active (NA), Input[4] is evaluated.
Input[4]	Input [4]	Class: Analog - The input with the lowest priority. Evaluated only if all other Inputs are NA, whereby any valid value is reflected at the Output.	-163.83 to 16383	If not active (NA), the Output is set to NA.

Table–5.238 Priority Input (4) Object Output Properties.

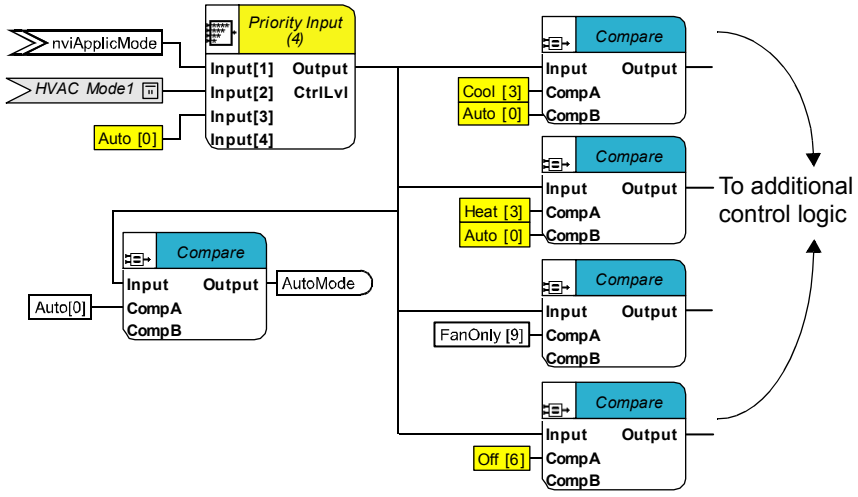
Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - Reflects the valid value at the highest priority input (Input[1] to Input[4]). A not active (NA) output results if all four inputs are NA.	-163.83 to 16383
CtrlLvl	Control Level	Class: Analog - Identifies the currently active input by priority number (1, 2, 3, or 4). If all inputs are not active (NA), this output is also NA.	1, 2, 3, or 4

Applying the Object

The Priority Input (4) object is typically used for “fallback” type logic within a control application. An example object is shown in Figure–5.123 below.

Example

In this example, Input[1] is an enumerated value from a network variable of a standard controller profile that allows the HVAC mode to be set from the network. When valid (if used) this value is always at the object’s output (feeding inputs of the five Compare objects). Input[2] is a similar enumerated value from a MicroNet sensor’s HVAC mode selection. This value appears at the output of the object only if Input[1] is not active (NA). Input[3] is a constant value 0, equivalent to an Auto in the enumerated value. This value appears at the Output of the Priority Input [4] object only if Inputs [1] and [2] are both NA. In this example, Input[4] is not used and is left unconnected.



Figure–5.123 Example Priority Input (4) Object.

Priority Value Select

Object Usage: The Priority Value Select object selects one of four Value inputs by using a high to low priority scan on four related digital inputs (Inputs[1] to [4]). The priority scan is from high (Input[1]) to low (Input[4]). The first ON found during this Input scan results in the value of the corresponding Value input (Values[1] to [4]) to be passed to the Output. If all digital inputs are either OFF or not active (NA), the output is set to a configuration Default Value.

Device Support: (all firmware revisions)

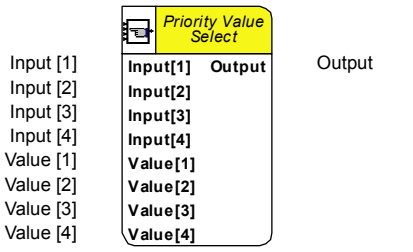
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 22 bytes
RAM: 24 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Default Value

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.239 Priority Value Select Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
Default	Default Value	Class: Analog - Defines the analog value produced at the output when all digital Inputs[1] to [4] are in either an OFF or not active (NA) state.	—	-163.83 to 16383 (or NA)	If desired, not active (NA) can be entered as the default value.

Table–5.240 Priority Value Select Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input[1]	Input [1]	Class: Digital - The input with the highest priority. An ON automatically passes the value of Value[1] to the Output.	—	If OFF or not active (NA), Input[2] is then evaluated.
Input[2]	Input [2]	Class: Digital - The input with the second highest priority. Evaluated only if Input[1] is OFF or NA. If ON, the value at Value[2] is passed to the Output.	—	If OFF or NA, Input[3] is then evaluated.
Input[3]	Input [3]	Class: Digital - The input with the third highest priority. Evaluated only if Inputs[1] and [2] are OFF or NA. If ON, the value at Value[3] is passed to the Output.	—	If OFF or NA, Input[4] is then evaluated.
Input[4]	Input [4]	Class: Digital - The input with the lowest priority. Evaluated only if Inputs[1], [2], [3] are OFF or NA. If ON, the value at Value[4] is passed to the Output.	—	If OFF or NA, the value of the Default Value is passed to the Output.
Value[1]	Value [1]	Class: Analog - The value passed to the Output whenever Input[1] is ON.	-163.83 to 16383	
Value[2]	Value [2]	Class: Analog - The value passed to the Output whenever Input[2] is ON and Input[1] is OFF or not active (NA).	-163.83 to 16383	
Value[3]	Value [3]	Class: Analog - The value passed to the Output whenever Input[3] is ON and Inputs[1] and [2] are OFF or not active (NA).	-163.83 to 16383	
Value[4]	Value [4]	Class: Analog - The value passed to the Output whenever Input[4] is ON and Inputs[1] [2], and [3] are OFF or not active (NA).	-163.83 to 16383	

Table–5.241 Priority Value Select Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - Reflects the value associated with the highest priority digital input (Inputs[1] to [4]) found to be in an ON state, or if all input are OFF/NA reflects the value in the “Default Value” property. An Output of not active (NA) results if the associated analog input (Values[1] to [4]) is NA, or if all Inputs[1] to [4] are OFF or NA and the Default Value property is set to NA.	-163.83 to 16383

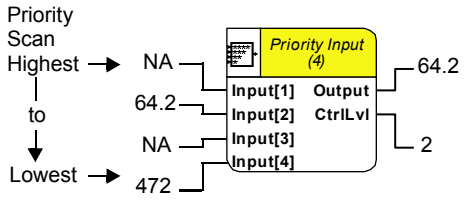
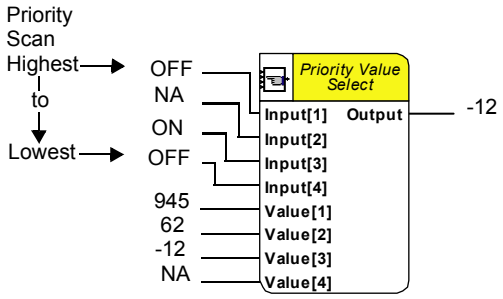
Applying the Object

The Priority Value Select object is similar to the Priority Input (4) object, as it provides a prioritized analog value from four possible analog inputs. However, it differs by providing additional digital inputs (Inputs[1] to [4]) as a means of value input selection. These inputs are scanned in a high Input[1] to low Input[4] fashion. The highest priority input found in an ON state determines which corresponding value input (Value[1] to [4]) is passed to the Output. Digital inputs that are OFF or not active (NA) are bypassed.

The Priority Value Select object also contains a Default Value configuration property. This assigned default value is passed to the Output whenever all four digital inputs are either in an OFF or NA state.

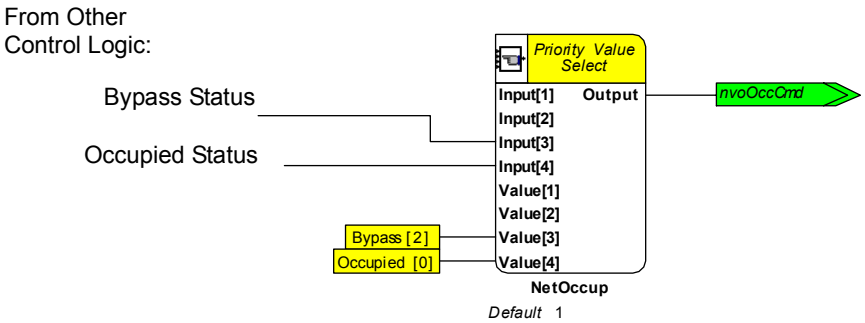
Priority Type Objects Compared

Both the Priority Input (4) object and the Priority Value Select object 'pass through' a value received on one of four inputs. The selection method for which input (value) is passed varies between the objects, as shown below.

Priority Input (4) object	Priority Value Select object
	
Passes through to the Output the first valid value (any value except NA) found on the continuous priority scan of the four Inputs[1] through [4], with Input[1] having the highest priority and Input[4] having the lowest priority. If all inputs have NA the Output is also NA. The currently passed input is indicated by the analog value on the Control Level output, from 1 to 4. (If all inputs are NA, the Control Level output is also NA.)	Passes through to the Output the value on a Value[1] through [4] input that corresponds to the first digital input (Inputs[1] through [4]) found ON during the continuous priority scan of the four digital inputs. Input[1] has the highest priority and Input[4] has the lowest priority. If a scanned input has an OFF or NA it's corresponding Value input is not used. The value passed may include NA if the Value input has an NA and its corresponding digital Input is ON. If Inputs[1] through [4] all have OFF or NA then a configurable Default Value is passed. This Default Value can be any value from -163.83 to 16383 or NA.

Priority Value Select Example

The Priority Value Select object used in this example generates a value required by the enumerated value type SNVT_occupancy, which is used in the network variable "nvoOccCmd". In this example, the Default Value is set to unoccupied (1) for when bypass and occupied conditions are not present.



Figure–5.124 Example Priority Value Select Object.

PWM

Object Usage: The Pulse Width Modulation (PWM) object is a point-type object with a physical hardware address for a digital output (DO). It behaves as a pulse width modulator by generating a time-proportioned ON / OFF control signal in response to a 0.0 to 100% input signal. The object is typically used for time-proportioned control valves, actuators, and electric heat loads. The cycle operation of the digital ON / OFF output is determined by the assigned time Period and the current Input value. The PWM object can also provide fixed or compensated duty-cycle control.

A PWM Priority object (page 427) is also available; it functions identically but with the addition of four (vs. one) prioritized inputs.

Device Support: (all firmware revisions)

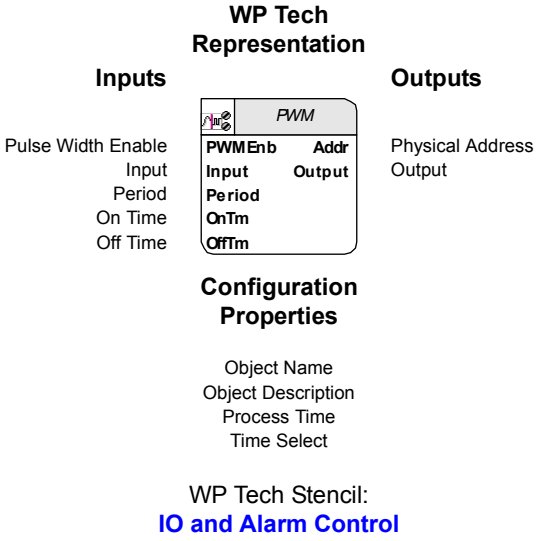
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 18 bytes
RAM: 24 bytes (standard controller)
6 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Time Select

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.242 PWM Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
TmSel	Time Select	Class: Analog Determines if the PWM resolution for time operation is in minutes or seconds.[a]	0 (Minutes)	0 - Minutes 1 - Seconds	A not active (NA) or value out of range results in a default of Minutes.

a. In MNL-11Rxx and MNL-13Rxx controller applications, whenever “Seconds” is selected for the Time Select property, the resolution is 0.1 sec. This accommodates wax motor applications, which require a higher resolution.

Table–5.243 PWM Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
PWMEnb	Pulse Width Enable	Class: Digital - An ON or not active (NA) enables the pulse width modulation function. An OFF disables the pulse width modulation function and holds the hardware (Output) in an OFF state.	—	
Input	Input	Class: Analog - The requested pulse width modulation demand (0.0 to 100.0%) used to determine the hardware (Output) action.	0.0 to 100.0%	A not active (NA) is evaluated as 0.0%.
Period	Period	Class: Analog - Defines the period or total time of each complete pulse width modulation cycle (in minutes or seconds, depending on the Time Select property). A zero, negative, or not active (NA) holds the hardware (Output) OFF.	3.0 to 1000.0 sec or 0.05 to 1000.0 min	Minutes or seconds, based on Time Select property[a].
OnTm	On Time	Class: Analog - The Minimum On time value in minutes or seconds, based on the property Time Select.	0.0 to 1000.0[b]	A negative value or NA disables the Minimum ON time function.
OffTm	Off Time	Class: Analog - The Minimum Off time value in minutes or seconds, based on the property Time Select.	0.0 to 1000.0[b]	A negative value or NA disables the Minimum OFF time function.

a. In MNL-11Rxx and MNL-13Rxx controller applications, whenever “Seconds” is selected for the Time Select property, the resolution is 0.1 sec. This accommodates wax motor applications, which require a higher resolution.
b. The sum of the On Time and Off Time inputs should be less than the value of the period input.

Table–5.244 PWM Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Addr	Physical Address	Class: Analog - Defines the physical hardware address (output terminal point on the controller) assigned to the PWM object.	Dependent on the controller platform selected.
Output	Output	Class: Digital - This output indicates the current digital output state corresponding to the pulse width modulation cycle.	OFF (0.0) ON (100.0)

Applying the Object

The PWM (Pulse Width Modulation) object provides a means for generating a time-proportioned Digital ON / OFF control signal from an input demand (0 to 100%) signal. Typical applications include time-proportioned control valves, actuators, and electric heat loads.

General Behavior

PWM Enable

The operation of the object is controlled by the PWM Enable input.

- An OFF disables the pulse width modulation function and sets the output to the OFF state. All timeout values in process are reset to their initial values, and the pulse width modulation cycle is reset to the beginning of the cycle (Period).
- An ON or not active (NA) enables the pulse width modulation function allowing the output to cycle at a calculated rate. The calculated rate is based upon the demand value applied to the Input.

Period and Input

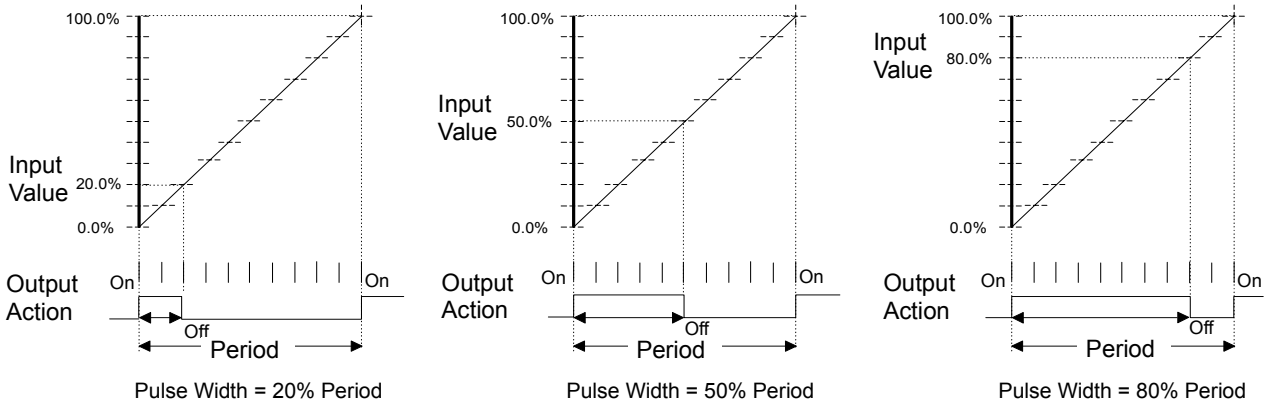
The value at the Period input establishes the PWM object's time base, or the repeating time for one complete ON and OFF output cycle. This value may range from 3.0 to 1000.0 seconds or 0.05 to 1000.0 minutes, depending on the setting of the Time Select configuration property.

The Input value range (0.0 to 100.0%) establishes the percentage of output ON time with respect to the assigned Period. Table–5.245 shows how the ON and OFF times at the hardware (Output) result from varying demand values at the Input.

Table–5.245 PWM Object Input to Output ON / OFF Times.

Input	Calculated Output ON time	Calculated Output OFF time
0.0%	Output held OFF	Output held OFF
10.0%	0.1 (Period)	0.9 (Period)
20.0%	0.2 (Period)	0.8 (Period)
30.0%	0.3 (Period)	0.7 (Period)
40.0%	0.4 (Period)	0.6 (Period)
50.0%	0.5 (Period)	0.5 (Period)
60.0%	0.6 (Period)	0.4 (Period)
70.0%	0.7 (Period)	0.3 (Period)
80.0%	0.8 (Period)	0.2 (Period)
90.0%	0.9 (Period)	0.1 (Period)
100.0%	Output held ON	Output held ON

As shown in Figure–5.125, when the Input equals a demand of 20.0% (at left) the Output cycles ON 20% of the Period and OFF 80% of the Period. This modulation is repeated as long as the demand remains at 20.0% and PWM Enable is set to ON. As the Input value changes to equal a demand of 50.0% (middle), the Output cycles ON 50% of the Period and OFF 50% of the Period. Again, this modulation repeats as long as the demand remains at 50.0% and PWM Enable is set to ON. As the Input value changes to equal a demand of 80% (at right), Output cycles ON 80% of the Period and OFF 20% of the Period. In this manner, the pulse width continually adjusts to changes at the Input.



Figure–5.125 PWM Object Pulse Width Modulation Varies from Input Value Change.

Applications and Examples

The PWM object can be used for time-proportioned control of two-position devices designed for ON / OFF time-proportioned control. Also, this object can be used in fixed duty cycle and compensated duty cycle applications.

Time-Proportioned Control

The PWM object provides a time-proportioned control output for the control of electric resistance heaters, two position spring return actuators, heat motor actuators, solenoid valves, etc. designed for ON / OFF time-proportioned control.

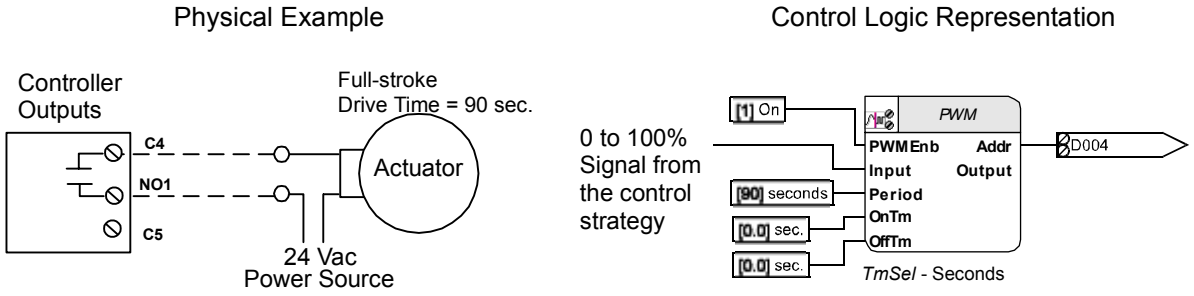
The value assigned for the Period is dependent upon the application.

- For two-position spring return actuators and wax motor actuators, the Period value should equal the full-stroke drive time of the actuator.
- For control of electric resistance heaters and solenoid valves, the Period value is dependent upon the application and the response of the equipment being controlled.

Note: To accommodate wax motor applications, the MNL-11Rxx and MNL-13Rxx controllers use a resolution of 0.1 sec whenever “Seconds” is selected for the PWM object’s Time Select property.

Basic PWM Operation

This example shows how a two-position spring return actuator is controlled by the PWM object. In this example, the Period is set to a value of 90 seconds, to match the specified actuator drive time. The On Time and Off Time values have been set to 0 (zero), as minimum ON and OFF time protection is not required for this application.



Figure–5.126 Basic PWM Operation

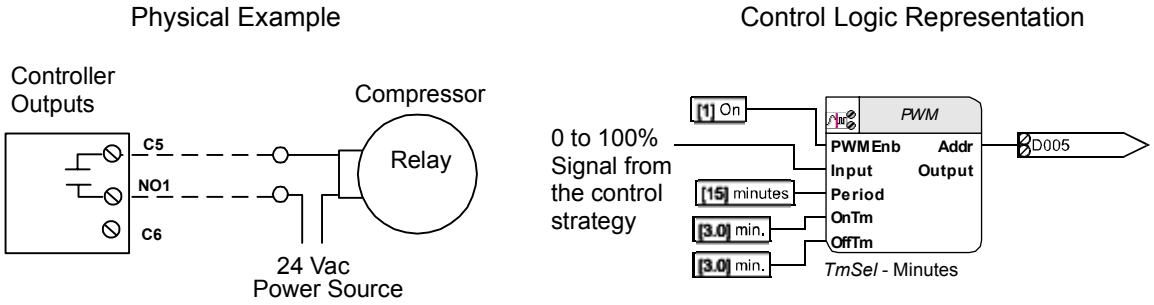
The output response from this example PWM object for a 90 Second period is shown below in Table–5.246.

Table–5.246 Example PWM Object Input to Output for 90 Second Period.

Input	Calculated Output ON time	Calculated Output OFF time
0.0%	Output held OFF	Output held OFF
10.0%	9.0 Seconds	81.0 Seconds
25.0%	22.5 Seconds	67.5 Seconds
33.3%	30.0 Seconds	60.0 Seconds
50.0%	45.0 Seconds	45.0 Seconds
66.6%	60.0 Seconds	30.0 Seconds
75.0%	67.5 Seconds	22.5 Seconds
90.0%	81.0 Seconds	9.0 Seconds
100.0%	Output held ON	Output held ON

Using Minimum On and Off Times

This example describes how a PWM object can be applied to equipment requiring Minimum ON and / or Minimum OFF short cycle protection. This example PWM object is setup to operate a compressor with a Period setting of 15 minutes and minimum on (On Time) and minimum off (Off Time) of 3 minutes.



Figure–5.127 Using Minimum On and Off Times

Based upon these settings, this compressor will typically cycle at a rate of 4 CPH (cycles per hour) whenever the input demand is within the active control range, where:

Active range CPH = 60 minutes ÷ Period in minutes

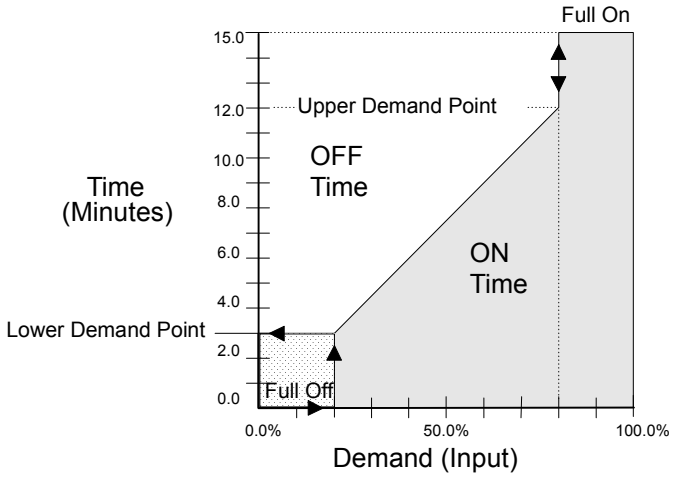
However, due to the dynamic response capability of the PWM object, the CPH is not limited to the active range number of cycles per hour. The PWM object allows for sudden changes in demand (Input) and responds to the changes as long as the Minimum On and Off time values have been satisfied. An ON cycle can be extended or terminated based upon the latest demand as long as the Minimum ON timeout has been satisfied. An OFF cycle can be extended or terminated based upon the latest demand as long as the Minimum OFF timeout has been satisfied.

Extreme changes to the demand signal can cause the cycles per hour to increase to a maximum CPH, which is determined by the values assigned to the Minimum On (On Time) and Minimum Off (Off Time) parameters, where:

CPH maximum = 60 minutes ÷ (On Time in minutes + Off Time in minutes)

For this example, while the typical active range CPH is 4, the actual maximum CPH is [60 minutes ÷ (3 min. On Time + 3 min. Off Time)], or 10 cycles per hour.

As the demand signal from the control strategy ranges between 0.0% and 100.0%, the PWM algorithm calculates the output ON / OFF action. Using the compressor parameters previously specified with an Input demand signal of 0.0%, the output is set to OFF.



Figure–5.128 Using Minimum On and Off Times: Input vs. Time Chart.

As the Input demand value increases, the output remains OFF until the demand equals or exceeds 20.0%. At this point, the Output is set to ON and all appropriate timers are initialized. The output remains ON for 3 minutes and returns to OFF for the remainder of the period or 12 minutes. This lower demand point is determined by dividing the minimum on time (On Time) by the assigned Period.

PWM Lower Demand Point: (minutes of On Time / Period)

In this example, the lower demand point is 3 minutes ÷ 15 minutes = 0.2, or 20.0%. As the demand (Input) value continues to increase, the calculated ON time increases proportionally to the Period. This continues until the demand (Input) value equals or exceeds 80.0%. At this point, the Output is set to ON and remains ON without cycling as the minimum off requirement can not be achieved within the total period.

The upper demand point is determined by dividing the Minimum Off time (Off Time) by the assigned Period and subtracting this result from the maximum demand or 100.0%.

Upper Demand Point: 100% - (minutes of Off Time / Period)

The upper demand point in this case is 100% - (3 minutes ÷ 15 minutes) = 0.8, or 80.0% The output remains full ON until the Input decreases to a demand value below 80.0% where cycling of the output can resume. As the demand (Input) value continues to decrease, the calculated ON time decreases proportionally to the period. This continues until the demand (Input) value equals 20.0% where the output is cycling at 3 minutes ON and 12 minutes OFF.

The 3 minute ON and 12 minute OFF cycle rate will remain constant as the demand (Input) continues below 20.0%, in order to maintain the equipment's Minimum On requirement. The output continues to cycle until the Input demand value equals 0.0%, at which time the output is held OFF. The output remains OFF until the demand (Input) equals or exceeds the lower demand point (20.0%), and the Minimum Off timeout has expired.

The output response from this example PWM object with a 15-minute Period and 3-minute Time On and Time Off values is shown in Table–5.247

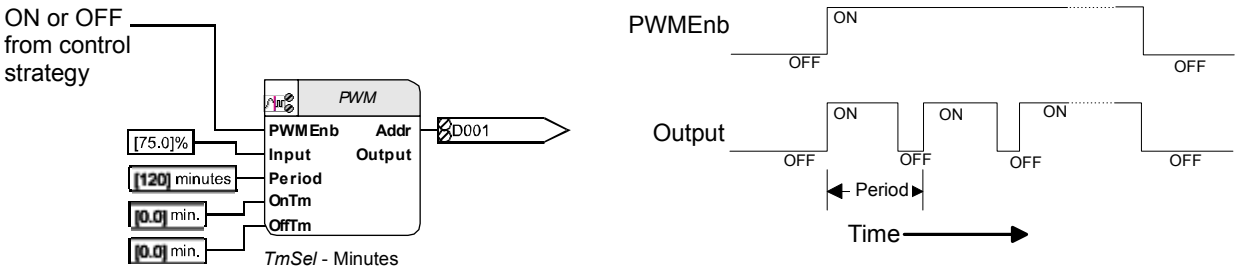
Table–5.247 Example PWM Object Input to Output Chart, Period = 15 minutes, Time On = 3 minutes and Time Off = 3 minutes.

Input	Calculated Output ON time	Calculated Output OFF time
0.0%	Output held OFF	Output held OFF
Less than 20.0%	From 0.0%, Output held OFF. or From > 20.0%, 3.0 minutes.	From 0.0%, Output held OFF. or From > 20.0%, 12.0 Minutes.
30.0%	4.5 Minutes	10.5 Minutes
40.0%	6.0 Minutes	9.0 Minutes
50.0%	7.5 Minutes	7.5 Minutes
60.0%	9.0 Minutes	6.0 Minutes
70.0%	10.5 Minutes	4.5 Minutes
80.0% to 100.0%	Output held ON	Output held ON

Fixed Duty Cycle Control

The PWM object can provide a fixed duty cycled digital output, or an output with a fixed ON time and a fixed OFF time. The total cycle period (ON time plus the OFF time) is determined by the Period value assigned.

The example in Figure–5.129 illustrates a fixed duty cycle based upon a 120.0 minute Period. A demand (Input) value of 75.0% causes the output to cycle at a rate of 90 minutes ON and 30 minutes OFF. (In this application, the Input is assigned a constant 75.0% value.) Fixed duty cycle control is initiated whenever the control strategy enables the PWM Enable input (OFF-to-ON transition at PWM Enable).

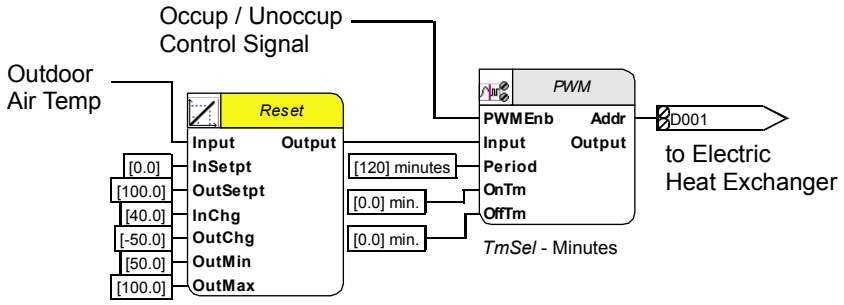


Figure–5.129 PWM Object Used for Fixed Duty Cycle Control.

The fixed duty cycle begins with the hardware (Output) set to ON when PWM Enable is set to ON. The output cycles at the preset frequency until the enable is returned to the OFF state. When PWM Enable is disabled (OFF), the hardware (Output) immediately returns to OFF regardless of present cycle conditions.

Compensated Duty Cycle Control

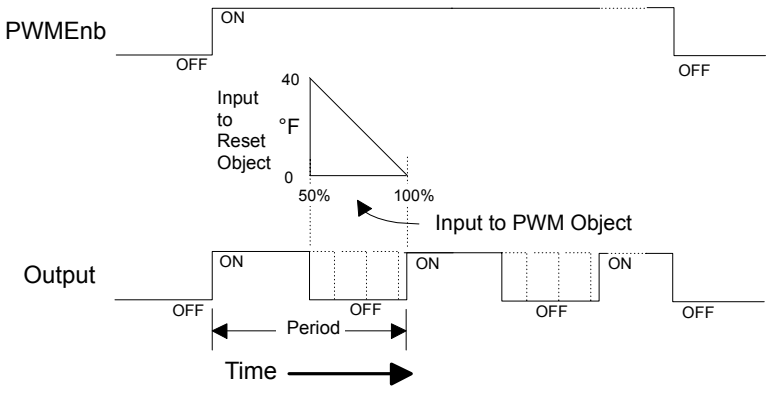
The PWM object can provide compensated duty cycle control when used with a Reset object that is configured to provide a compensation variable for the ON portion of the PWM cycle. Figure–5.130 below illustrates how the PWM object can be used to duty cycle the heating mode of an electric controlled heat exchanger.



Figure–5.130 PWM Object Used in a Compensated Duty Cycle Application.

In this case, the heating mode is enabled by the Occupied/Unoccupied control signal which enables the compensated duty cycle during the occupied period of the day. Through the use of the PWM object, energy usage is limited to a controlled portion of a two hour period. The mass of the controlled media and the typical heat loss rate are such that the system can coast for approximately one hour before the loss of heat becomes noticeable. The sizing of the heating equipment is such that the heat loss can be recovered and maintained within one hour.

However, as the outdoor air temperature drops below 40°F, the losses become noticeable and the equipment needs to operate longer to maintain space temperature. When the outdoor air temperature drops to 0°F, it is necessary for the heating equipment to be ON 100% of the time, as shown below in Figure–5.131.



Figure–5.131 Example Reset Action for PWM Compensated Duty Cycle Control.

With the use of the Reset object, and the compensated duty cycle feature, the time is increased where the heating is enabled as the outdoor air temperature decreases until the duty cycle feature is totally removed from the sequence of operation.

PWM Priority

Object Usage: Like the PWM object, the PWM Priority object is a point-type object with a physical hardware address for a digital output (DO) that behaves as a pulse width modulator. This object differs from the PWM object in that it features four prioritized inputs instead of a single input (but otherwise works the same). Pulse width modulation is done by generating a time-proportioned ON / OFF control signal in response to the active priority 0.0 to 100% input signal. This object is typically used for time-proportioned control valves, actuators, and electric heat loads. The cycle operation of the digital ON / OFF output is determined by the assigned time Period and the active Input value. As with the PWM object, the PWM Priority object can provide fixed or compensated duty cycle control.

Device Support: (all firmware revisions)

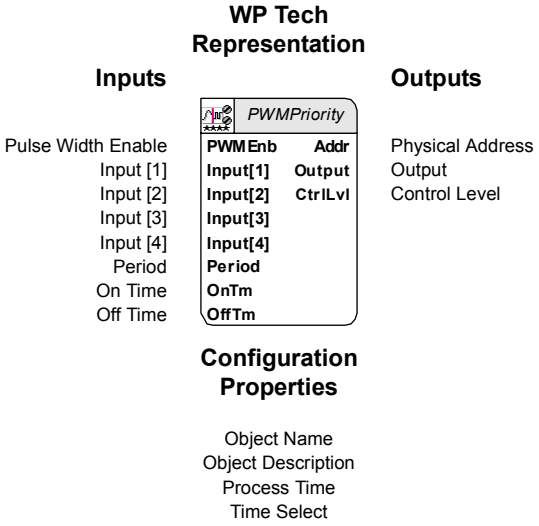
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 24 bytes
RAM: 32 bytes (standard controller)
8 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Time Select

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.248 PWM Priority Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.248 PWM Priority Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
TmSel	Time Select	Class: Analog - Determines if the PWM resolution for time operation is in minutes or seconds.[a]	0 (Minutes)	0 - Minutes 1 - Seconds	A not active (NA) or value out of range results in a default of Minutes.

a. In MNL-11Rxx and MNL-13Rxx controller applications, whenever “Seconds” is selected for the Time Select property, the resolution is 0.1 sec. This accommodates wax motor applications, which require a higher resolution.

Table–5.249 PWM Priority Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
PWMEnb	Pulse Width Enable	Class: Digital - An ON or not active (NA) enables the pulse width modulation function. An OFF disables the pulse width modulation function and holds the hardware (Output) in an OFF state.	—	
Input[1]	Input[1]	Class: Analog - The requested pulse width modulation demand with the highest priority. This input is monitored first to control the physical and logical object outputs.	0.0 to 100.0%	A not active (NA) at this input causes the second input to be evaluated for a valid value.
Input[2]	Input[2]	Class: Analog - The requested pulse width modulation demand with the second highest priority. This input is monitored if Input[1] has a NA, and is used to control the physical and logical object outputs.	0.0 to 100.0%	A not active (NA) at this input causes the third input to be evaluated for a valid value.
Input[3]	Input[3]	Class: Analog - The requested pulse width modulation demand with the third highest priority. This input is monitored if Inputs[1] and [2] are both NA, and is used to control the physical and logical object outputs.	0.0 to 100.0%	A not active (NA) at this input causes the fourth and last input to be evaluated for a valid value.
Input[4]	Input[4]	Class: Analog - The requested pulse width modulation demand with the lowest priority.This input is monitored if all other Inputs have a not active (NA), and is used to control the physical and logical object outputs.	0.0 to 100.0%	If all inputs including Input[4] have a not active (NA), the hardware and logical output are OFF as 0.0% demand is assumed.
Period	Period	Class: Analog - Defines the period or total time of each complete pulse width modulation cycle (in minutes or seconds, depending on the Time Select property). A negative or not active (NA) holds the hardware (Output) OFF.	3.0 to 1000.0 sec or 0.05 to 1000.0 min	Minutes or seconds, based on Time Select property[a].
OnTm	On Time	Class: Analog - The Minimum On time value in minutes or seconds, based on the property Time Select.	0.0 to 1000.0[b]	A negative or NA value disables the Minimum ON time function.
OffTm	Off Time	Class: Analog - The Minimum Off time value in minutes or seconds, based on the property Time Select.	0.0 to 1000.0[b]	A negative or NA value disables the Minimum OFF time function.

a. In MNL-11Rxx and MNL-13Rxx controller applications, whenever “Seconds” is selected for the Time Select property, the resolution is 0.1 sec. This accommodates wax motor applications, which require a higher resolution.

b. The sum of the On Time and Off Time inputs should be less than the value of the Period input.

Table–5.250 PWM Priority Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Addr	Physical Address	Class: Analog - Defines the physical hardware address (output terminal point on the controller) assigned to the PWM object.	Dependent on the controller platform selected.
Output	Output	Class: Digital - This output indicates the current digital output state corresponding to the pulse width modulation cycle.	OFF (0.0) ON (100.0)
CtrlLvl	Control Level	Class: Analog - Indicates the currently active input by providing the priority number of the related input, that is 1, 2, 3, or 4. If all four inputs have a not active (NA), this output also goes to NA.	1, 2, 3, or 4

Applying the Object

The PWM (Pulse Width Modulation) Priority object provides a means for generating a time-proportioned Digital ON / OFF control signal from a prioritized input signal representing demand (0 to 100%). Typical applications include time-proportioned control valves, actuators, and electric heat loads.

General Behavior

PWM Enable

The operation of the object is controlled by the PWM Enable input.

- An OFF disables the pulse width modulation function and sets the output to the OFF state. All timeout values in process are reset to their initial values, and the pulse width modulation cycle is reset to the beginning of the cycle (Period).
- An ON or not active (NA) enables the pulse width modulation function allowing the output to cycle at a calculated rate. The calculated rate is based upon the demand value applied to the Input.

Period

The value at the Period input establishes the PWM Priority object's time base, or the repeating time for one complete ON and OFF output cycle. This value may range from 3.0 to 1000.0 seconds or 0.05 to 1000.0 minutes, depending on the setting of the Time Select configuration property. The highest priority Input[1 - 4] value (0.0 to 100.0%) establishes the percentage of output ON time with respect to the assigned Period. Table–5.251 shows how the ON and OFF times at the hardware (Output) result from varying demand values at the highest priority active Input.

Table–5.251 PWM Priority Object Input to Output ON / OFF Times.

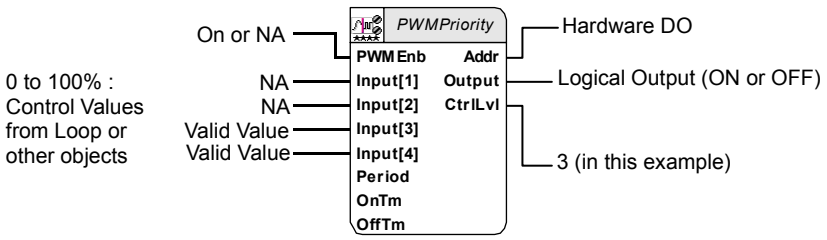
Highest Active Priority Input	Calculated Output ON time	Calculated Output OFF time
0.0%	Output held OFF	Output held OFF
10.0%	0.1 (Period)	0.9 (Period)
20.0%	0.2 (Period)	0.8 (Period)
30.0%	0.3 (Period)	0.7 (Period)
40.0%	0.4 (Period)	0.6 (Period)
50.0%	0.5 (Period)	0.5 (Period)
60.0%	0.6 (Period)	0.4 (Period)
70.0%	0.7 (Period)	0.3 (Period)

Table–5.251 PWM Priority Object Input to Output ON / OFF Times.

Highest Active Priority Input	Calculated Output ON time	Calculated Output OFF time
80.0%	0.8 (Period)	0.2 (Period)
90.0%	0.9 (Period)	0.1 (Period)
100.0%	Output held ON	Output held ON

Priority Inputs and Values

Input[1] is the highest priority input, and is always evaluated first on each scan of the inputs. Any valid value present on Input[1] becomes the Input to the object, regardless of the state of the other inputs. A valid value is any numeric value besides a not active [NA].

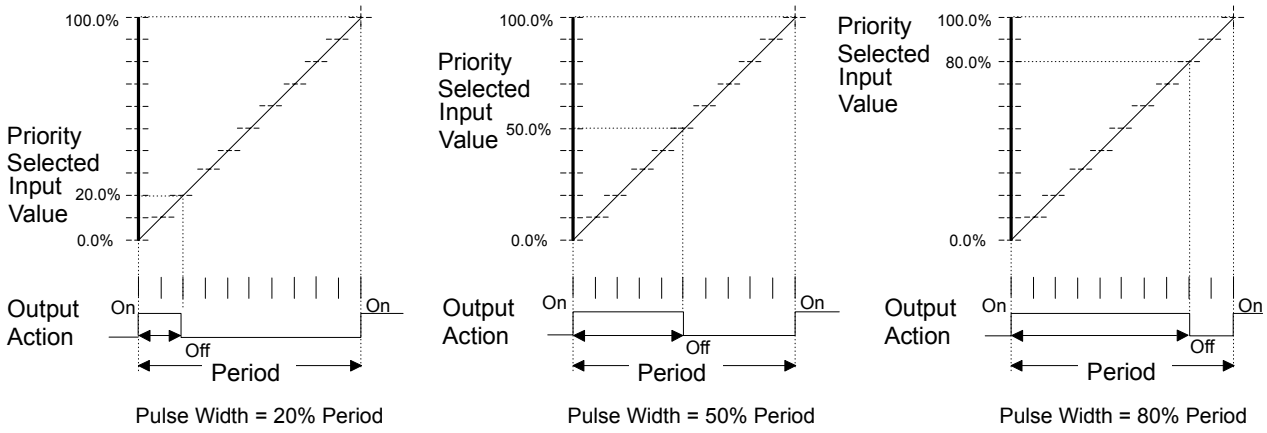


Figure–5.132 Input[3] as the Current Active Input.

If Input[1] has an NA, then Input[2] is evaluated in the same manner. This priority scan continues only if Input[2] also has an NA, at which point Input[3] is evaluated, and if Input[3] also has an NA, to lastly evaluate Input[4]. If Input[4] also has an NA, then the hardware and logical Outputs are held in an OFF state, and the Control Level output indicates NA.

Typically, input values are within a normal range, that is, between 0.0 and 100.0%. However, any value outside this range is evaluated as either 0.0 or 100.0. For example, a value of 165.0 is evaluated as 100.0. Likewise, a negative value such as - 56.7 would be evaluated by the object as 0.0.

Figure–5.133 shows when the highest priority Input equals a demand of 20.0% (at left), the Output cycles ON 20% of the Period and OFF 80% of the Period. This modulation is repeated as long as the demand remains at 20.0% and PWM Enable is set to ON. As the priority Input value changes to equal a demand of 50.0% (middle), the Output cycles ON 50% of the Period and OFF 50% of the Period.



Figure–5.133 PWM Priority Object Pulse Width Modulation Varies from Priority Selected Input Value Change.

Again, this modulation repeats as long as the demand remains at 50.0% and PWM Enable is set to ON. As the priority Input value changes to equal a demand of 80% (at right), Output cycles ON 80% of the Period and OFF 20% of the Period. The pulse width continually adjusts to changes at the highest priority Input.

Applications and Examples

The PWM Priority object can be used for time-proportioned control of two-position devices for time-proportioned ON / OFF control. Also, this object can be used in fixed duty-cycle and compensated duty-cycle applications.

Time-Proportioned Control

The PWM Priority object provides a time-proportioned control output for the control of electric resistance heaters, two position spring return actuators, heat motor actuators, solenoid valves, etc. designed for time-proportioned ON / OFF control.

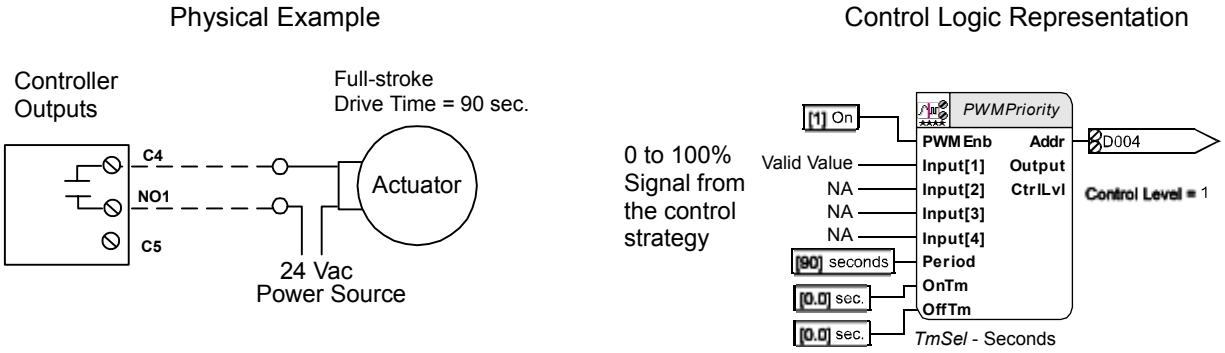
The value assigned for the Period is dependent upon the application.

- For two-position spring return actuators and wax motor actuators, the Period value should equal the full-stroke drive time of the actuator.
- For control of electric resistance heaters and solenoid valves, the Period value is dependent upon the application and the response of the equipment being controlled.

Note: To accommodate wax motor applications, the MNL-11Rxx and MNL-13Rxx controllers use a resolution of 0.1 sec whenever “Seconds” is selected for the PWM object’s Time Select property.

Basic PWM Operation

This example shows how a two-position spring return actuator is controlled by the PWM Priority object. In this example, the Period is set to a value of 90 seconds, to match the specified actuator drive time. The On Time and Off Time values have been set to 0 (zero), as minimum ON and OFF time protection is not required for this application.



Figure–5.134 Basic PWM Operation

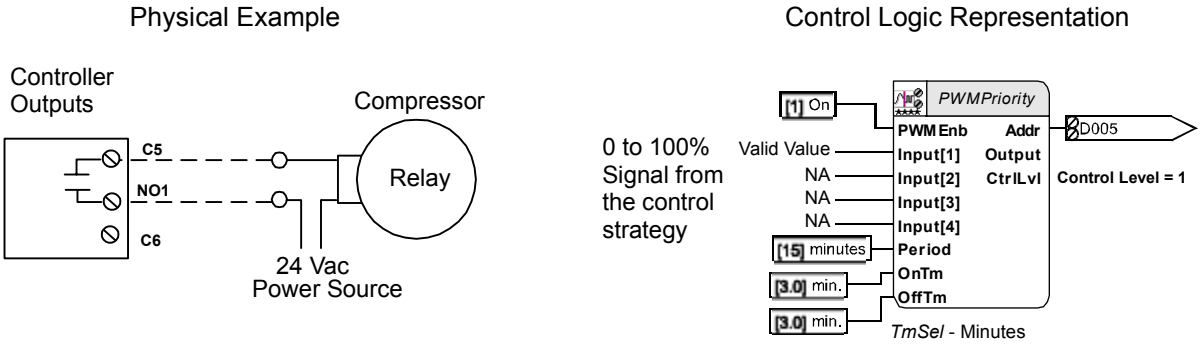
The output response from this example is shown in Table–5.252 below.

Table–5.252 Example PWM Priority Object for 90 Second Period.

Highest Active Priority Input	Calculated Output ON time	Calculated Output OFF time
0.0%	Output held OFF	Output held OFF
10.0%	9.0 Seconds	81.0 Seconds
25.0%	22.5 Seconds	67.5 Seconds
33.3%	30.0 Seconds	60.0 Seconds
50.0%	45.0 Seconds	45.0 Seconds
66.6%	60.0 Seconds	30.0 Seconds
75.0%	67.5 Seconds	22.5 Seconds
90.0%	81.0 Seconds	9.0 Seconds
100.0%	Output held ON	Output held ON

Using Minimum On and Off Times

This example describes how a PWM Priority object can be applied to equipment requiring Minimum ON and / or Minimum OFF short cycle protection. This example PWM Priority object is setup to operate a compressor with a Period setting of 15 minutes and minimum on (On Time) and minimum off (Off Time) of 3 minutes.



Figure–5.135 Using Minimum on and Off Times

Based upon these settings, this compressor will typically cycle at a rate of 4 CPH (cycles per hour) whenever the prioritized Input demand is within the active control range, where:

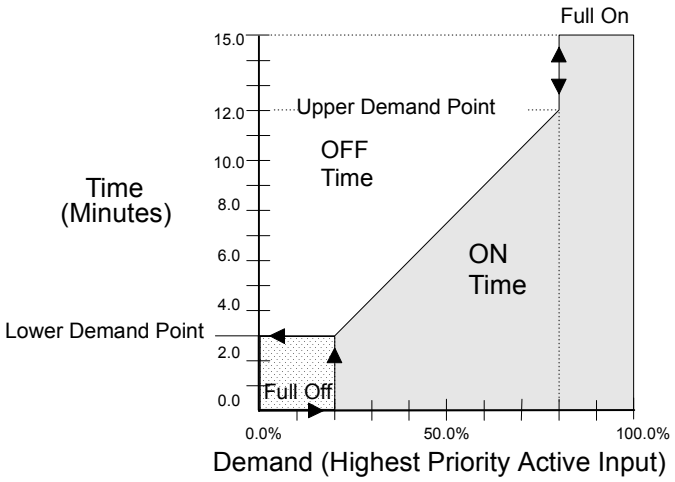
Active range CPH = 60 minutes ÷ Period in minutes

However, due to the dynamic response capability of the PWM Priority object, the CPH is not limited to the active range number of cycles per hour. The PWM Priority object allows for sudden changes in demand (prioritized Input) and responds to the changes as long as the Minimum On and Off time values have been satisfied. An ON cycle can be extended or terminated based upon the latest demand as long as the Minimum ON timeout has been satisfied. An OFF cycle can be extended or terminated based upon the latest demand as long as the Minimum OFF timeout has been satisfied. Extreme changes to the demand signal can cause the cycles per hour to increase to a maximum CPH, which is determined by the values assigned to the Minimum On (On Time) and Minimum Off (Off Time) parameters, where:

CPH maximum = 60 minutes ÷ (On Time in minutes + Off Time in minutes)

For this example, while the typical active range CPH is 4, the actual maximum CPH is [60 minutes ÷ (3 min. On Time + 3 min. Off Time)], or 10 cycles per hour.

As the prioritized Input demand signal from the control strategy ranges between 0.0% and 100.0%, the PWM Priority algorithm calculates the output ON / OFF action. Using the compressor parameters previously specified with a demand signal of 0.0%, the output is set to OFF.



Figure–5.136 Using Minimum On and Off Times: Prioritized Input vs. Time Chart.

As the prioritized Input demand value increases, the output remains OFF until the demand equals or exceeds 20.0%. At this point, the Output is set to ON and all appropriate timers are initialized. The output remains ON for 3 minutes and returns OFF for the remainder of the period or 12 minutes. This lower demand point is determined by dividing the minimum on time (On Time) by the assigned Period.

PWM Lower Demand Point: (minutes of On Time / Period)

In this example, the lower demand point is 3 minutes ÷ 15 minutes = 0.2, or 20.0%. As the prioritized Input value (demand) continues to increase, the calculated ON time increases proportionally to the Period. This continues until the demand value equals or exceeds 80.0%. At this point, the Output is set to ON and remains ON without cycling as the minimum off requirement can not be achieved within the total period. The upper demand point is determined by dividing the Minimum Off time (Off Time) by the assigned Period and subtracting this result from the maximum demand or 100.0%.

Upper Demand Point: 100% - (minutes of Off Time / Period)

The upper demand point in this case is 100% - (3 minutes ÷ 15 minutes) = 0.8, or 80.0% The output remains full ON until the prioritized Input value (demand) decreases to a value below 80.0%, where cycling of the output can resume. As the demand value continues to decrease, the calculated ON time decreases proportionally to the period. This continues until the demand value equals 20.0% where the output is cycling at 3 minutes ON and 12 minutes OFF.

The 3 minute ON and 12 minute OFF cycle rate will remain constant as the demand continues below 20.0%, in order to maintain the equipment's Minimum On requirement. The output continues to cycle until the demand value equals 0.0%, at which time the output is held OFF. The output remains OFF until the demand equals or exceeds the lower demand point (20.0%), and the Minimum Off timeout has expired.

The output response from this example PWM Priority object with a 15 minute Period and 3 minute Time On and Time Off values is shown in Table–5.253.

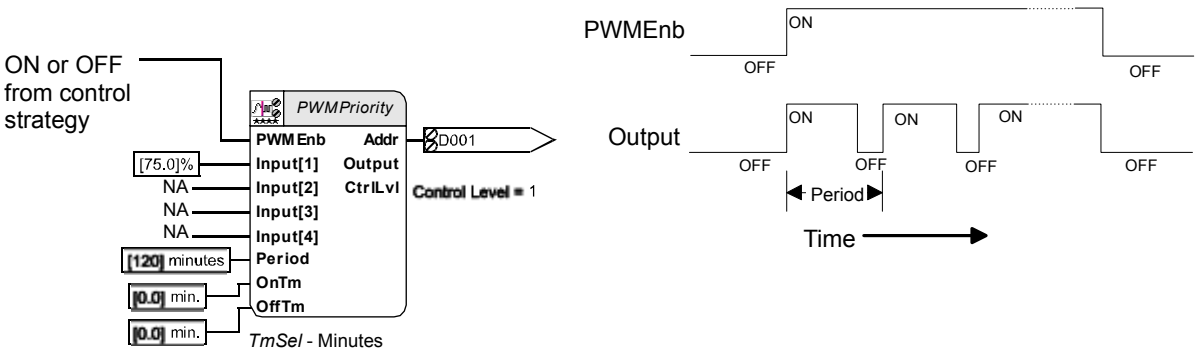
Table–5.253 Example PWM Priority Object Input to Output Chart, Example 2.

Highest Active Priority Input	Calculated Output ON time	Calculated Output OFF time
0.0%	Output held OFF	Output held OFF
Less than 20.0%	From 0.0%, Output held OFF. or From > 20.0%, 3.0 minutes.	From 0.0%, Output held OFF. or From > 20.0%, 12.0 Minutes.
30.0%	4.5 Minutes	10.5 Minutes
40.0%	6.0 Minutes	9.0 Minutes
50.0%	7.5 Minutes	7.5 Minutes
60.0%	9.0 Minutes	6.0 Minutes
70.0%	10.5 Minutes	4.5 Minutes
80.0% to 100.0%	Output held ON	Output held ON

Fixed Duty Cycle Control

The PWM Priority object can provide a fixed duty cycled digital output, or an output with a fixed ON time and a fixed OFF time. The total cycle period (ON time plus the OFF time) is determined by the Period value assigned.

Figure–5.137 illustrates a fixed duty cycle based upon a 120.0 minute Period. A demand (prioritized Input) value of 75.0% causes the output to cycle at a rate of 90 minutes ON and 30 minutes OFF. (In this application, only Input[1] is used, and is assigned a constant 75.0% value.) Fixed duty cycle control is initiated whenever the control strategy enables the PWM Enable input (OFF-to-ON transition at PWM Enable).

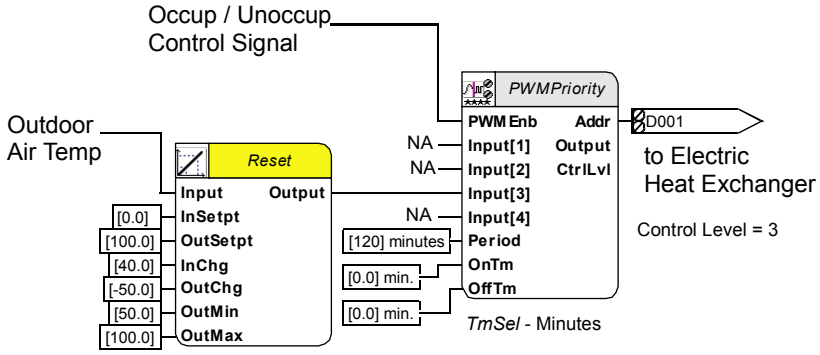


Figure–5.137 PWM Priority Object Used for Fixed Duty Cycle Control.

The fixed duty cycle begins with the hardware (Output) set to ON when PWM Enable is set to ON. The output cycles at the preset frequency until the enable is returned to the OFF state. When PWM Enable is disabled (OFF), the hardware (Output) immediately returns to OFF regardless of present cycle conditions.

Compensated Duty Cycle Control

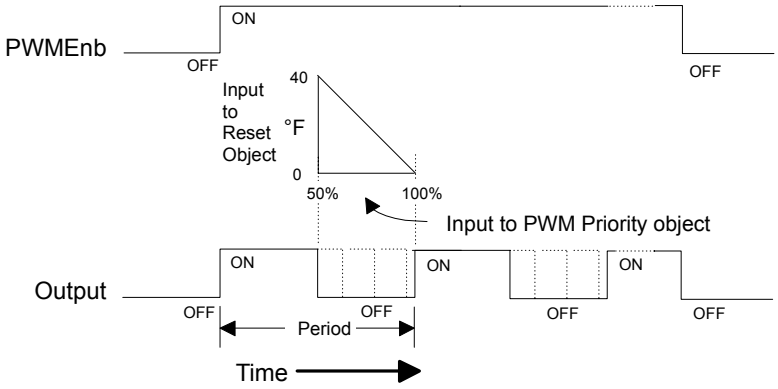
The PWM Priority object can provide compensated duty cycle control when used with a Reset object that is configured to provide compensation variable for the ON portion of the PWM cycle. Figure–5.138 illustrates how the PWM Priority object can be used to duty cycle the heating mode of an electric controlled heat exchanger.



Figure–5.138 PWM Priority Object in a Compensated Duty Cycle Application.

In this case, the heating mode is enabled by the Occupied/Unoccupied control signal which enables the compensated duty cycle during the occupied period of the day. Through the use of the PWM Priority object, energy usage is limited to a controlled portion of a two hour period. The mass of the controlled media and the typical heat loss rate are such that the system can coast for approximately one hour before the loss of heat becomes noticeable. The sizing of the heating equipment is such that the heat loss can be recovered and maintained within one hour.

However, as the outdoor air temperature drops below 40°F, losses become noticeable and the equipment needs to operate longer to maintain space temperature. When outdoor air temperature drops to 0°F, the heating equipment must be ON 100% of the time, as shown in Figure–5.139.



Figure–5.139 Example Reset Action for PWM Compensated Duty Cycle Control.

With the use of the Reset object, and the compensated duty cycle feature, the time is increased where the heating is enabled as the outdoor air temperature decreases until the duty cycle feature is totally removed from the sequence of operation.

Ramp

Object Usage: The Ramp object provides a way for increasing or decreasing an analog output value at a user-specified rate. The object can be configured to perform two different styles of ramp control:

- Standard Analog Ramp - Increases or decreases the output at a slope and rate defined by the output minimum, output maximum, and time values.
- Step Change Ramp - Incrementally increases or decreases the output at the level and rate defined by the step level and time values.

The ramp Time input accepts any value between 0 and 10,000 seconds. The standard analog ramp function uses the Time input to calculate ramp rate or speed. The step change ramp function uses the Time input to determine the output step period.

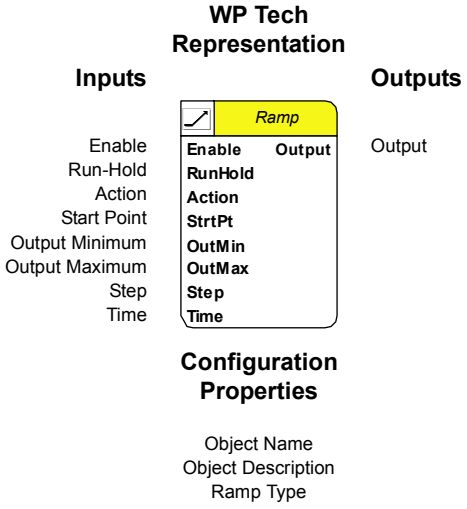
Device Support:

MN 800 series

Memory Requirements: (per object)

EEPROM: 22 bytes
RAM: 10 bytes

WP Tech Representation



Configuration Properties

Object Name
Object Description
Ramp Type

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.254 Ramp Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
RmpTyp	Ramp Type	Class: Analog - Defines the type of ramp algorithm used by the object, either standard analog (0) or step change (1).	0	Analog (0) Step Change (1)	Not Active (NA) or values outside range result as analog (0).

Table–5.255 Ramp Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Enable	Enable	Class: Digital - An ON or not active (NA) allows the Ramp algorithm to update the output (ramp or step accordingly). An OFF disables the Ramp algorithm, causing the output to be set and held at the value present on the Start Point input.	—	
RunHold	Run-Hold	Class: Digital - An OFF freezes the Output, causing it to be held at the currently calculated value. An ON or not active (NA) allows the output to update based on the normal operation of the standard analog ramp function or the step change ramp function. In the case of a step change ramp, a hold-to-run (OFF-to-ON) transition will immediately step the output and initiate the step timer.	—	A not active (NA) causes the output to be NA.
Action	Action	Class: Digital - An OFF or not active (NA) causes the output to be direct-acting, meaning the output value increases during the ramping function. An ON causes the output to be reverse-acting, meaning the output value decreases during the ramping function.	—	
StrtPt	Start Point	Class: Analog - Defines the output value whenever an OFF is at the Enable input. Any valid value is accepted. The output is released from the Start Point value whenever the object is enabled.	-163.83 to 16383	A not active (NA) causes the output to be NA.
OutMin	Output Minimum	Class: Analog - Defines the minimum output value during an active ramp or step operation. Typically less than Output Maximum. If Output Minimum is set to a value greater than the Output Maximum, the Ramp algorithm outputs the Output Maximum value.	-163.83 to 16383	A not active (NA) causes the output to be NA.
OutMax	Output Maximum	Class: Analog - Defines the maximum output value during an active ramp or step operation. Typically greater than Output Minimum. If Output Maximum is set to a value less than the Output Minimum, the Ramp algorithm outputs the Output Maximum value.	-163.83 to 16383	A not active (NA) causes the output to be NA.
Step	Step	Class: Analog - Evaluated only if the Ramp object is configured for a step change ramp function. For a step ramp change function, this must be a positive number. The output is adjusted in increments defined by the Step value each step period, in the direction defined by the Action input, as long as the Run-Hold input is in the run condition. *Note:* The first step adjustment occurs immediately whenever a hold-to-run transition is detected.	0 to 16383	A not active (NA) causes the Step value to default to 0 (zero). No step change to the output occurs as a result.
Time	Time	Class: Analog - Defines the ramp rate (Standard Analog) or step period (Step Change) as follows: • Standard Analog Ramp - Defines the ramp rate as the amount of time required to ramp between the output minimum and maximum. • Step Change Ramp - Used to define the duration of time between output steps (step period). With either ramp function, a value of 0 (zero) causes the Ramp algorithm to output the Output Minimum or Maximum value based upon the requested direction.	0 to 10000 seconds	A not active (NA) disables the Ramp object, causing the Output to be set and held to the Start Point value.

Table–5.256 Ramp Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - Provides the value of the present analog ramp or step change ramp. This output is not active (NA) whenever any of these inputs have a NA: Start Point, Output Minimum, Output Maximum.	-163.83 to 16383

Applying the Object

The Ramp object is configured with the Ramp Type configuration property, to be either a standard analog ramp or a step change ramp. Analog ramps are typically used for the soft starting of equipment that is modulated with a continuous analog signal. The two types of ramps are explained separately.

Standard Analog Ramp

The Ramp object performs the standard analog ramp function when the Ramp Type configuration property is set to Standard Analog (0).

Enable and disable of the Ramp object is done at the Enable input. An OFF at Enable disables the Ramp object, causing the output to be set and held to the value at the Start Point input. An Enable of ON or not active (NA) enables the Ramp object, allowing the output to operate (ramp) as follows, based on the settings of the applicable inputs:

Run-Hold: The Run-Hold input provides a way to suspend or resume the ramping function:

- An OFF at Run-Hold causes the output to be held at the currently calculated output value (Hold).
- An ON or not active (NA) allows the ramp algorithm to run and update the output based upon the direction, slope, and rate defined (Run).

Action: The Action input determines the direction of the ramp:

- An OFF or not active (NA) at Action causes the output to be direct-acting, meaning the output value increases during the ramping function.
- An ON at Action causes the output to be reverse-acting, meaning the output value decreases during the ramping function.

Start Point: The Ramp object output is set to the Start Point value whenever the Enable input has an OFF state. This can be any valid Start Point value. A Start Point of not active (NA) causes the output to be set to NA. The output is released from the Start Point value whenever the Ramp object is enabled. Upon release, the output ramps or holds depending upon the value at the Run-Hold input. The release point or the value where the output is initialized is based upon the relationship of the Start Point to the Output Minimum and Output Maximum values.

A Start Point value between Output Minimum and Output Maximum will release and ramp from the Start Point value. A Start Point value less than the Output Minimum will release from the Output Minimum value and ramp accordingly. A Start Point value greater than the Output Maximum will release from the Output Maximum value and ramp accordingly.

Output Minimum: The Output Minimum defines the minimum value allowed at the output during ramp operation. This can be any valid Start Point value. An Output Minimum of not active (NA) sets the output to NA. If Output Minimum is at a greater value than the Output Maximum, the Ramp algorithm sets the output to the value at the Output Maximum.

Output Maximum: The Output Maximum defines the maximum value allowed at the output during ramp operation. This can be any valid Start Point value. An Output Maximum of not active (NA) sets the output to NA. If Output Maximum is a smaller value than the Output Minimum, the Ramp algorithm sets the output to the value at the Output Maximum.

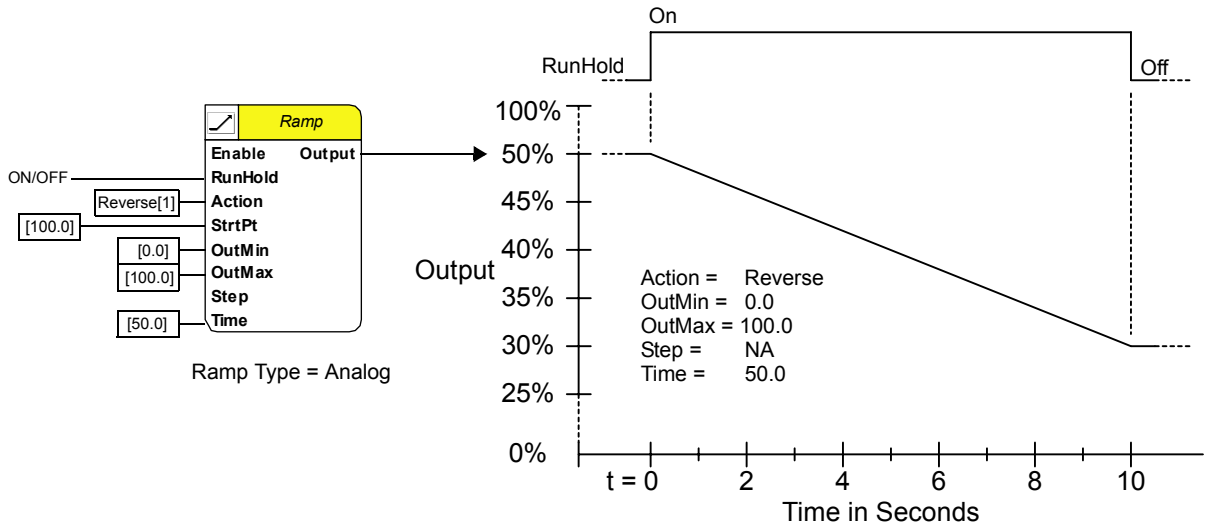
Step: In a standard analog ramp function, the Step input value is ignored.

Time: The standard analog ramp algorithm uses the Time input to determine the ramp rate or speed. The Time input can be any value between 0 and 10,000 seconds, which represents the amount of time for the output to ramp between the minimum and maximum output values.

Enabling and running the ramp with an assigned time value of zero causes the algorithm to set the output to the Minimum or Maximum value, based upon the requested ramp direction. A time value of not active (NA) disables the Ramp object, causing the Output to be set and held to the Start Point value.

Example Analog Ramp

The Ramp object in Figure–5.140 is configured for an analog ramp output, with the output produced as shown.



Figure–5.140 Example Ramp Object as Analog Ramp

The ramp output rate of change (change per second or cps) can be calculated as follows:

Output rate cps = (Output Maximum - Output Minimum) ÷ Time

In the example above, Output Minimum = 0.0%, Output Maximum = 100.0%, and Time = 50.0 seconds. Therefore:

Output rate cps = (100 - 0) ÷ 50 seconds

Output rate cps = 2.0% per second

The output decreases at a rate of 2.0% for every second the RunHold input is held in the Digital ON state. This example shows that the output will ramp linearly from 50% to 30% in a ten-second timeframe. Returning the RunHold input to Digital OFF causes the output to be held at the last calculated value.

Step Change Ramp

The Ramp object performs the step change ramp function when the Ramp Type configuration property is set to Step Change (1).

Enable and disable of the Ramp object is done at the Enable input. An OFF at Enable disables the Ramp object, causing the output to be set and held to the value at the Start Point input. An Enable of ON or not active (NA) enables the Ramp object, allowing the output to operate (step) as follows, based on the settings of the applicable inputs:

Run-Hold: The Run-Hold input provides a way to suspend or resume the ramping function:

- An OFF at Run-Hold causes the output to be held at the currently calculated output value (Hold).
- An ON or not active (NA) allows the ramp algorithm to run and update the output based upon the direction, step, and time defined (Run).

Action: The Action input determines the direction of the step:

- An OFF or not active (NA) at Action causes the output to be direct-acting, meaning the output value increases during the ramping function.
- An ON at Action causes the output to be reverse-acting, meaning the output value decreases during the ramping function.

Start Point: The Ramp object output is set to the Start Point value whenever the Enable input has an OFF state. This can be any valid Start Point value. A Start Point of not active (NA) causes the output to be set to NA. The output is released from the Start Point value whenever the Ramp object is enabled. Upon release, the output steps or holds depending upon the value at the Run-Hold input. The release point or the value where the output is initialized is based upon the relationship of the Start Point to the Output Minimum and Output Maximum values.

A Start Point value between Output Minimum and Output Maximum will release and step from the Start Point value. A Start Point value less than the Output Minimum will release from the Output Minimum value and step accordingly. A Start Point value greater than the Output Maximum will release from the Output Maximum value and step accordingly.

Output Minimum: The Output Minimum defines the minimum value allowed at the output during the ramp step operation. This can be any valid Start Point value. An Output Minimum of not active (NA) sets the output to NA. If Output Minimum is at a greater value than the Output Maximum, the Ramp algorithm sets the output to the value at the Output Maximum.

Output Maximum: The Output Maximum defines the maximum value allowed at the output during the ramp step operation. This can be any valid Start Point value. An Output Maximum of not active (NA) sets the output to NA. If Output Maximum is a smaller value than the Output Minimum, the Ramp algorithm sets the output to the value at the Output Maximum.

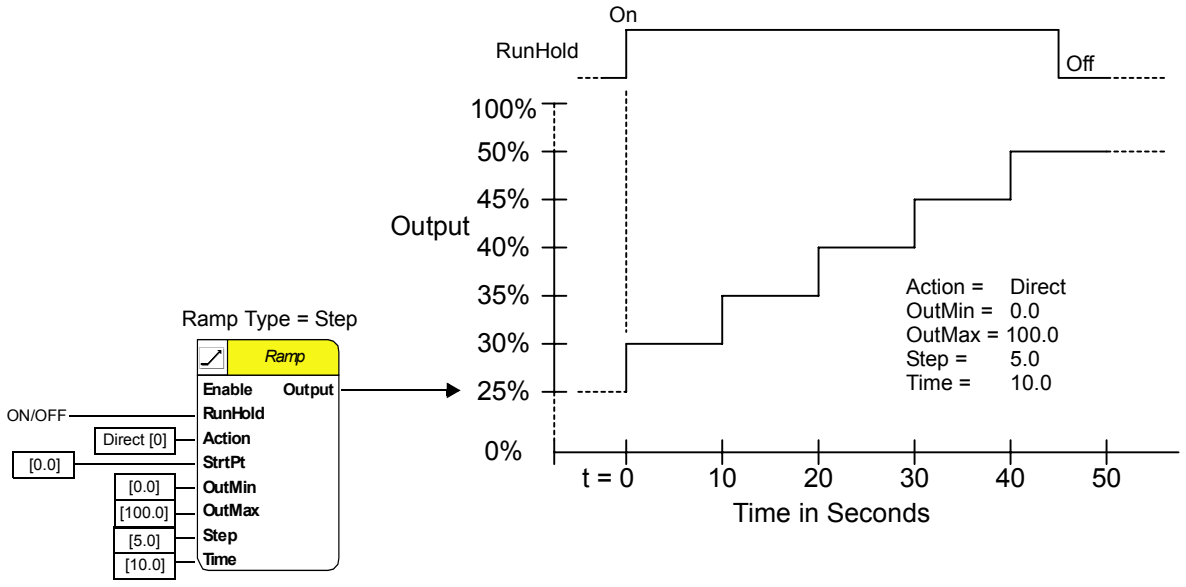
Step: For the step change ramp function, the Step value must be a positive number. A Step value of not active (NA) is defaulted to 0 (zero). The output is adjusted by the Step value assigned in the direction defined by the Action input, either incremented (direct action) or decremented (reverse action). The output continues to be adjusted each time period, as long as the Run-Hold input is set to Run (ON).

Note: The first step adjustment is made immediately whenever a transition from hold-to-run is detected.

Time: The step change ramp algorithm uses the Time input to determine the output step period, that is, the interval between each step change at the output. The Time input can be process any value between 0 and 10,000 seconds. Enabling and running the step change ramp with an assigned time value of zero causes the algorithm to set the output to the Minimum or Maximum value, based upon the requested step direction. A time value of not active (NA) disables the Ramp object, causing the Output to be set and held to the Start Point value.

Example Step Change Ramp

The Ramp object in Figure–5.140 is configured for a step change ramp output, with the output produced as shown.



Figure–5.141 Example Ramp Object as Step Change Ramp.

In the example above, the output steps at a rate of 5.0% in every ten-second period as long as the Run-Hold input is in the ON state. The example shows the output steps from 25% to 50% in a forty-second timeframe. Returning the Run-Hold input to OFF causes the output to be held at the last calculated value.

Reset

Object Usage: The Reset object provides a proportional and limited output for setpoint adjustment or reset of a control loop, based on a changing independent variable input. For example, the Reset object can calculate a new boiler control setpoint, based on a change in outdoor air temperature. The reset function also provides for maximum and minimum limit values on the calculated output.

Device Support: (all firmware revisions)

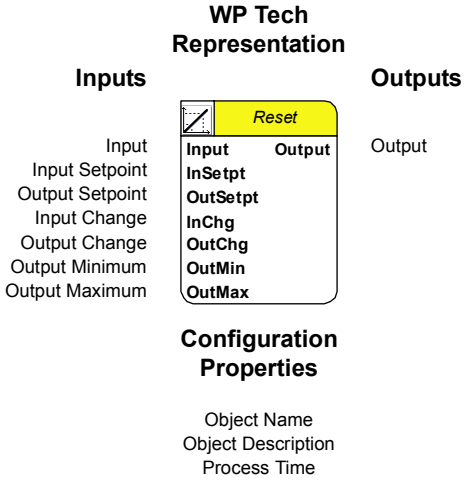
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 18 bytes
RAM: 20 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Loop and Process Control

Properties

Table–5.257 Reset Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.258 Reset Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input	Input	Class: Analog - The sensed value of the media being controlled. This value is compared to the Input Setpoint value and is used by the reset algorithm to calculate the Output value.	-163.83 to 16383	If not active (NA), the Output is set to NA.

Table–5.258 Reset Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selection	Notes
InSetpt	Input Setpoint	Class: Analog - The input setpoint value, where the Output value equals the Output Setpoint.	-163.83 to 16383	If not active (NA), the Output is set to NA.
OutSetpt	Output Setpoint	Class: Analog - The output setpoint value. The Output equals the Output Setpoint when the Input value equals the Input Setpoint.	-163.83 to 16383	If not active (NA), the Output is set to NA.
InChg	Input Change	Class: Analog - The value and direction of input change required to cause the Output to change the amount defined by the Output Change value.	-163.83 to 16383	If not active (NA), the Output is set to NA.
OutChg	Output Change	Class: Analog - The value and direction of output change. The Output changes by this amount when the Input changes by the Input Change value.	-163.83 to 16383	If not active (NA), the Output is set to NA.
OutMin	Output Minimum	Class: Analog - Defines the minimum value allowed at the Output.	-163.83 to 16383	If not active (NA), the Output has no minimum limit.
OutMax	Output Maximum	Class: Analog - Defines the maximum value allowed at the Output.	-163.83 to 16383	If not active (NA), the Output has no maximum limit.

Table–5.259 Reset Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The calculated output reset value. A not active (NA) indicates one (or more) of the following has a NA: Input, Input Setpoint, Output Setpoint, Input Change, Output Change.	-163.83 to 16383

Applying the Object

The Reset object provides the ability to change a setpoint (control point) of a control loop, thermostat, etc., based on a change of another variable. The reset calculation uses a proportional ratio based on the values assigned to the inputs Output Setpoint, Input Setpoint, Output Change, and Input Change. The range of Output can be limited by values assigned to the inputs Output Minimum and Output Maximum. Depending on the Input Change and Output Change values, the reset may be either direct acting or reverse acting.

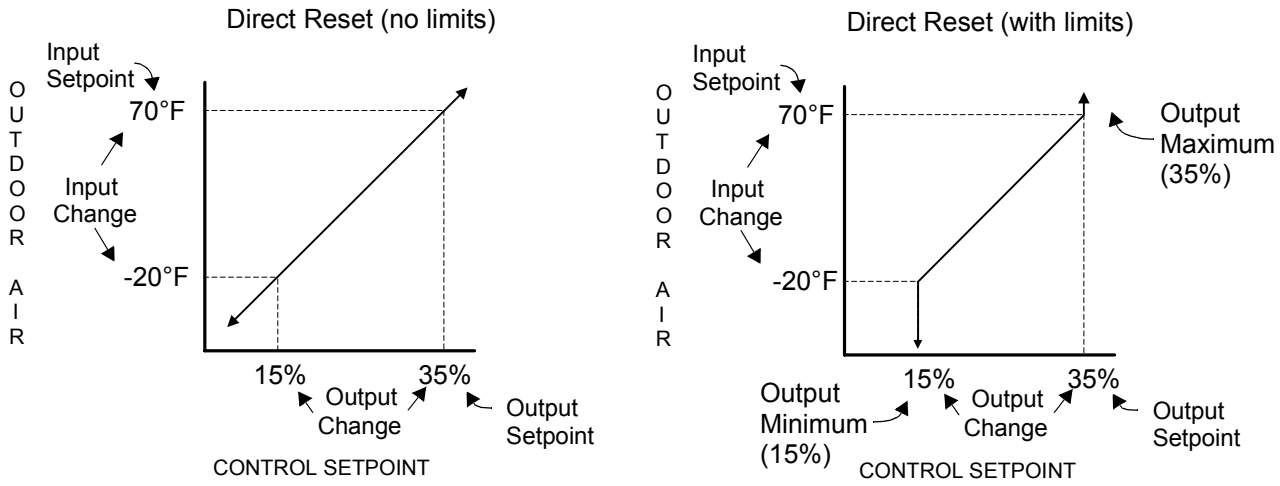
- Direct reset means that an increasing Input value produces an increasing Output value.
- Reverse reset means that an increasing Input value produces a decreasing Output value.

Direct Reset Example

The setpoint of a humidity controlling application is to be reset by the outdoor air temperature. As the outdoor air temperature decreases from 70°F to -20°F, the humidity setpoint is to change (be reset) from 35% to 15%. The reset control is direct because, as the outdoor temperature decreases, the humidity setpoint is decreased.

Using this information, the Reset object properties can be determined. At an input value of 70°F, the output value required is 35%. Therefore, the Input Setpoint can be set to 70 and the Output Setpoint can be set to 35. Next, the Input Change and Output Change values must be determined.

Input Change = 70°F - (-20°F) Answer = 90
Output Change = 35% - 15% Answer = 20



Figure–5.142 Example Direct Reset With and Without Output Limits.

If desired, Output limits may be set by assigning values to the Output Minimum and Output Maximum inputs. Without these limits, the Output can continue above the Output Setpoint (35%) or below the Output Setpoint - Output Change (15%), as shown on the left side of Figure–5.142.

In this example, a Minimum Output value of 15% and a Maximum Output value of 35% limits the Output within this range regardless of outdoor temperature, as shown on the right side of Figure–5.142.

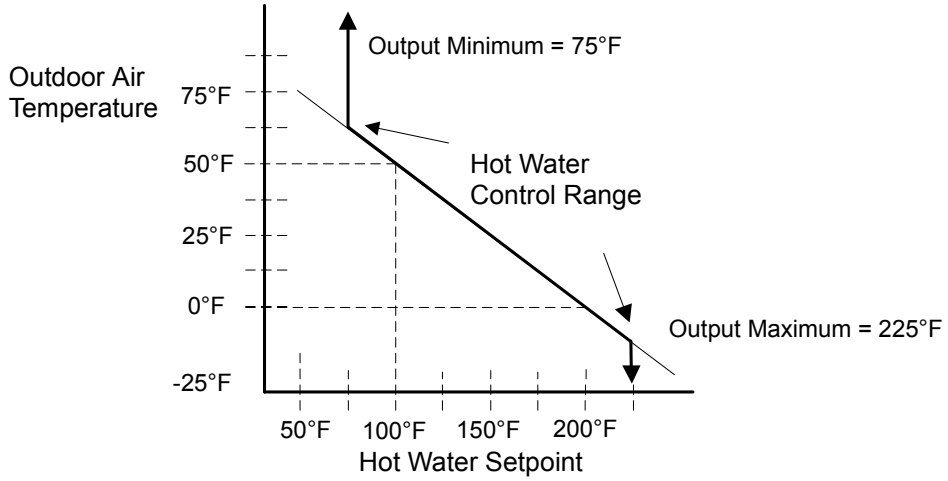
Reverse Reset Example

In this example, the Reset object will reset the setpoint of a hot water control application, based on the outdoor air temperature. When the outdoor air temperature is 0°F, the hot water setpoint is 200°F. As the outdoor air increases to 50°F, the hot water setpoint is reset to 100°F.
This reset is reverse because, as the outdoor air temperature increases, the hot water setpoint is decreased.

Using this information, the Reset object properties can be determined.
At an input value of 0°F, the output value required is 200°F. Therefore, the Input Setpoint can be set to 0 and the Output Setpoint can be set to 200.
Next, the Input Change and Output Change values must be determined.

Input Change = 0°F - (50°F) Answer = -50
Output Change= 200°F - 100°F Answer = 100

The negative Input Change of -50 results in the required reverse reset.



Figure–5.143 Example Reverse Reset With Output Limits

In this example, the hot water setpoint is limited by the Output Minimum and Output Maximum values. The Output Minimum is set to 75°F and the Output Maximum is set to 225°F. The calculated output hot water setpoint value cannot exceed these limits, regardless of outdoor air temperature.

Reset Ratio

The Reset object uses the Input Change and Output Change values to calculate a ratio which is used by the reset algorithm.

Reset Ratio = Output Change ÷ Input Change

If Reset Ratio > 0, then Reset = DIRECT reset.

If Reset Ratio < 0, then Reset = REVERSE reset.

The output is calculated as follows:

Output = (Reset Ratio x (Input - Input Setpt)) + Output Setpt

The Output is limited between OutMin and OutMax.

Note:

- Input Change and Output Change values should not be set to zero. Setting Input Change and/or Output Change to zero will result in the following:

Table–5.260 Results of Input and/or Output Change Set to Zero.

Input Change	Output Change	Output Set To
0	0	Output Setpoint Value
Valid Value	0	Output Setpoint Value
0	Negative Value	Output Maximum Value
0	Positive Value	Output Minimum Value

- Setting Output Maximum to a value less than the Output Minimum value results in an Output value equal to the Output Maximum. Conversely, setting Output Minimum to a value greater than the Output Maximum value results in an Output value equal to the Output Maximum.

Using Known Reset Ratios

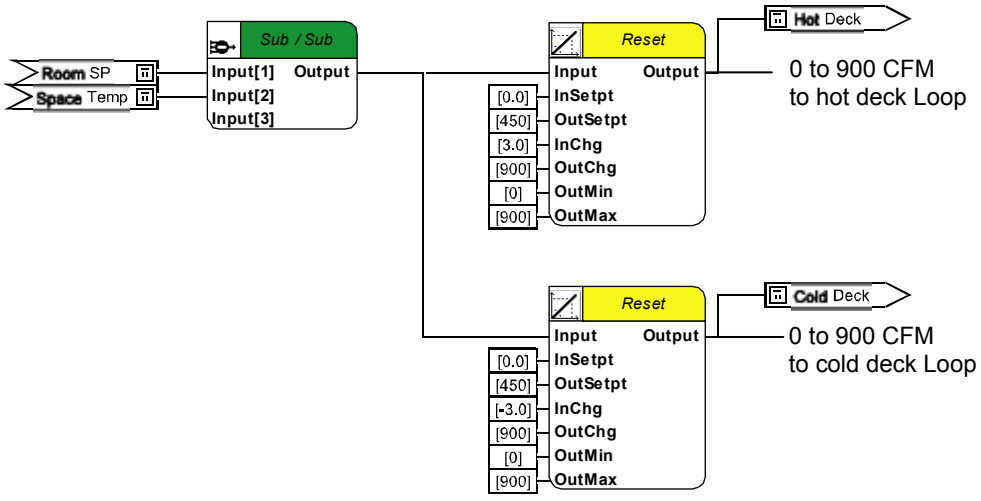
A known reset ratio can be entered directly for a Reset object, without calculating the Output Change value and Input Change value. The Input Setpoint and Output Setpoint values are still required.

To use a known reset ratio directly: set the Output Change value to the known ratio, and set the Input Change value to +1.0.

- Output Change value is positive (+X.X) for DIRECT reset.
- Output Change value is negative (-X.X) for REVERSE reset.

Control Logic Example

Two Reset objects are used in the application shown below in Figure–5.144. The application purpose is to reset the CFM setpoints of a constant volume roof top unit (RTU), which has both a hot deck and a cold deck. The Input to the Reset objects is the difference between the current room temperature setpoint and the current space temperature, calculated by a Sub / Sub math object. If this signal is positive, a heating demand is indicated; if this signal is negative, a cooling demand is indicated. If this signal is 0 (zero), space temperature is at setpoint.



Figure–5.144 Example Reset Object Control Application.

Correspondingly, the hot deck Reset object has a positive Input Change (InChg) assignment (3.0°F) and the cold deck Reset object has a negative Input Change assignment (-3.0°F). Each Reset object outputs a CFM setpoint from 0 to 900 CFM, with a 450 CFM output held at setpoint. The combined outputs of the two Reset objects always equals 900 CFM.

Schedule 7-Day

Object Usage: The 7-Day Schedule object provides a means for scheduling a seven day, repeating set of events. Each day can be programmed to provide up to four scheduled events. In addition, this object provides exception handling for up to four conditions. The 7-Day Schedule object, in conjunction with the Calendar object, provides a complete solution for yearly schedule control.

The 7-Day Schedule object is customized at the time it is copied onto the drawing page, so that its Current output is set to a numeric value, an on/off value, a SNVT_occupancy enumeration value, or an unoccupied/occupied value.

The 7-Day Schedule algorithm dynamically calculates all outputs based upon numerous control conditions, including schedule enable, forced override, temporary overrides, exceptions, and regular schedule assignments. The algorithm is designed to handle all schedule needs including scheduled operations that maintain an event through midnight.

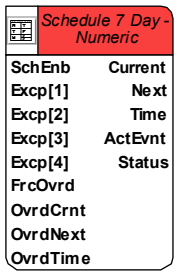
Device Support:

MN 800 series

Memory Requirements: (per object)

EEPROM: 206 bytes
RAM: 14 bytes

WP Tech Representation



Inputs

Schedule Enable
Exception 1
Exception 2
Exception 3
Exception 4
Force Override
Override Current
Override Next
Override Time

Outputs

Current
Next
Time
Active Event
Control Status

Numeric Type
Schedule 7-Day Object Shown

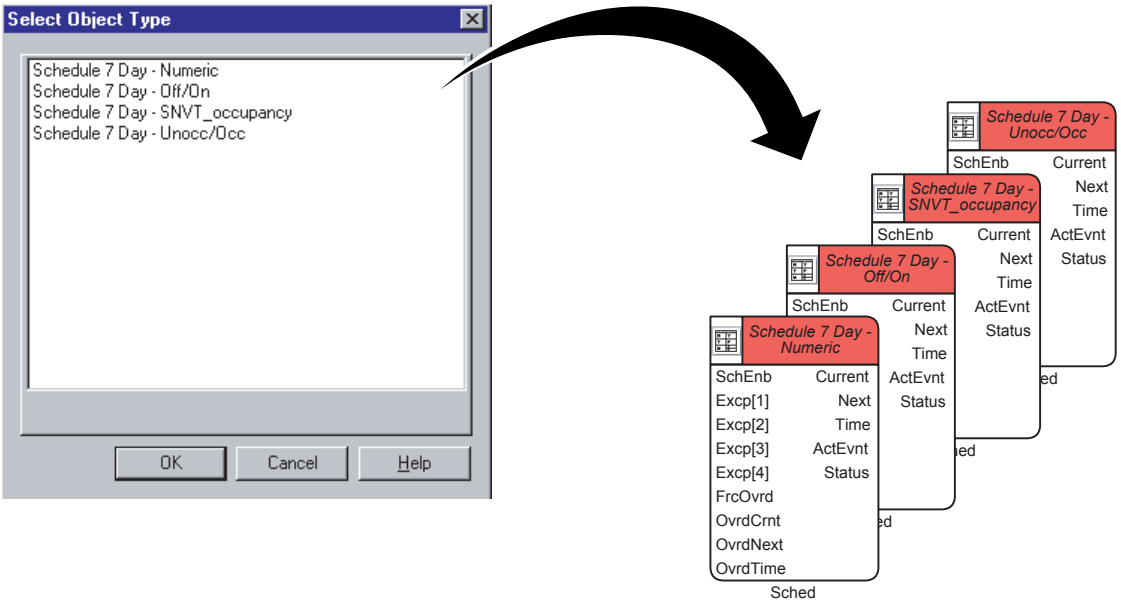
Configuration Properties

Object Name
Object Description
Event Format
Default Current
Default Next
Default Time
Sun Sched [1] to [4] / Event [1] to [4]
Mon Sched [1] to [4] / Event [1] to [4]
Tue Sched [1] to [4] / Event [1] to [4]
Wed Sched [1] to [4] / Event [1] to [4]
Thu Sched [1] to [4] / Event [1] to [4]
Fri Sched [1] to [4] / Event [1] to [4]
Sat Sched [1] to [4] / Event [1] to [4]
Excep 1 Sched [1] to [4] / Event [1] to [4]
Excep 2 Sched [1] to [4] / Event [1] to [4]
Excep 3 Sched [1] to [4] / Event [1] to [4]
Excep 4 Sched [1] to [4] / Event [1] to [4]

WP Tech Stencil:
Schedule Control

Adding a Schedule 7-Day Object to a Drawing

Adding a Schedule 7-Day object to the application causes a Select Object Type window to appear, allowing selection of the schedule's Event Format Figure-5.145. The Event Format defines the data format WP Tech uses when displaying the event selections in the object property editor.



Figure–5.145 Selection of 7-Day Schedule Object Types.

The Event Format selection defines the functionality of the object's Current and Next outputs:

- If set to Numeric Value — Events are entered as a value that can range from -163.83 to 16383 and not active (NA). The Current output will be set to the assigned numeric value for each scheduled event.
- If set to Off / On — Events are selected from a list consisting of off (0) and On (100). The Current output is typically set to On for the active event and Off for the inactive event.
- If set for SNVT_occupancy — Events are selected from an enumerated list that reflects the occupancy SNVT. The Current output is set to the assigned SNVT enumeration value for each scheduled event. SNVT_occupancy is defined as follows:

Occupied (0)	Standby (3)
Unoccupied (1)	Not Active (255)
Bypass (2)	

For more information on SNVT_occupancy, refer to the "MN Controllers SNVTs Quick Reference" section in Appendix B of this manual, on page 663.

- If set to Unoccupied / Occupied — Events are selected from a list consisting of Unoccupied (0) and Occupied (100). The Current output is typically set to Occupied for the active event and Unoccupied for the inactive event.

Properties

Table–5.261 Schedule 7-Day Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
DefCrnt	Default Current	Class: Analog — Defines the value assigned to the Current output whenever default conditions exist.	—	Based upon the Event Format selected. For value definitions, refer to Notes given for the Event Format property.	Default conditions exist whenever the object is disabled and / or there is an invalid controller time. A Force Override has priority over the default as long as the Schedule object remains enabled.
DefNext	Default Next	Class: Analog — Defines the value assigned to the Next output whenever default conditions exist.	—		
DefTime	Default Time	Class: Analog — Defines the value assigned to the Time output whenever default conditions exist.	—	-163.83 to 16383 and (NA)	

Table–5.261 Schedule 7-Day Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
SunSch[1] to SunSch[4] SunEvnt[1] to SunEvnt[4]	*Sunday* Schedule [1] to Schedule [4] Event [1] to Event [4]	Schedules consist of seven days, with each day allowing the assignment of one to four schedule times and events. **Schedule Times** Class: Time — Schedule time values are defined in hours and minutes (HH:MM). This object's algorithm uses the time (hours) value to determine and calculate the schedule operation. **Schedule Events** Class: Analog — Defines the events associated with each regular schedule program time. The event is based upon the Event Format selected. For value definitions, refer to Notes given for the Event Format property.	—	HH:MM Where: HH = 00 to 23 (Hours value) MM = 00 to 59 (Minutes value) Defined by Event Format configuration selections, including: • Numeric Value • Off/On • Occupied/ Unoccupied • Enumeration value defined by Echelon's SNVT_occupancy.	A schedule time is considered invalid whenever the assigned time is set to a not active (NA) or to any time which is outside the predefined hour / minute range. Daily scheduled times must be programmed in ascending chronological order for proper operation. An invalid schedule time, or a time value that is out of sequence, will cause the particular schedule event to be bypassed or ignored. The schedule object will run at least the first valid schedule time within the active day.
MonSch[1] to MonSch[4] MonEvnt[1] to MonEvnt[4]	*Monday* Schedule [1] to Schedule [4] Event [1] to Event [4]				
TueSch[1] to TueSch[4] TueEvnt[1] to TueEvnt[4]	*Tuesday* Schedule [1] to Schedule [4] Event [1] to Event [4]				
WedSch[1] to WedSch[4] WedEvnt[1] to WedEvnt[4]	*Wednesday* Schedule [1] to Schedule [4] Event [1] to Event [4]				
ThuSch[1] to ThuSch[4] ThuEvnt[1] to ThuEvnt[4]	*Thursday* Schedule [1] to Schedule [4] Event [1] to Event [4]				
FriSch[1] to FriSch[4] FriEvnt[1] to FriEvnt[4]	*Friday* Schedule [1] to Schedule [4] Event [1] to Event [4]				
SatSch[1] to SatSch[4] SatEvnt[1] to SatEvnt[4]	*Saturday* Schedule [1] to Schedule [4] Event [1] to Event [4]				

Table–5.261 Schedule 7-Day Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Exp1Sch[1] to Exp1Sch[4] Exp1Evnt[1] to Exp1Evnt[4]	*Exception 1* Schedule [1] to Schedule [4] Event [1] to Event [4]	Exception schedules consist of four exceptions, with each exception day allowing the assignment of up to four schedule times and events. **Exception Times** Class: Time — Exception schedule time values are defined in hours and minutes (HH: MM). **Exception Events** Class: Analog — Defines the events associated with each exception schedule time. The event is based upon the Event Format selected. For value definitions, refer to Notes given for the Event Format property.	—	HH:MM Where: HH = 00 to 23 (Hours value) MM = 00 to 59 (Minutes value) Defined by Event Format configuration selections, including Numeric Value, Off/On, Occupied/Unoccupied, and an enumeration value defined by Echelon's SNVT_occupancy.	A schedule time is considered invalid whenever the assigned time is set to a not active (NA) or to any time which is outside the predefined hour / minute range. Exception scheduled times must be programmed in ascending chronological order for proper operation. An invalid schedule time or a time value that is out of sequence will cause the particular schedule event to be bypassed or ignored. The schedule object will run at least the first valid schedule time within the active exception schedule. Exception schedule event values must follow the same format used to program the regular schedule events. This format is defined by the Event Format configuration property.
Exp2Sch[1] to Exp2Sch[4] Exp2Evnt[1] to Exp2Evnt[4]	*Exception 2* Schedule [1] to Schedule [4] Event [1] to Event [4]				
Exp3Sch[1] to Exp3Sch[4] Exp3Evnt[1] to Exp3Evnt[4]	*Exception 3* Schedule [1] to Schedule [4] Event [1] to Event [4]				
Exp4Sch[1] to Exp4Sch[4] Exp4Evnt[1] to Exp4Evnt[4]	*Exception 4* Schedule [1] to Schedule [4] Event [1] to Event [4]				

Table–5.262 Schedule 7-Day Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
SchEnb	Schedule Enable	Class: Digital — A Schedule Enable input of Digital ON or not active (NA) enables the 7-Day Schedule function. A Schedule Enable input of Digital OFF disables the 7-Day Schedule, causing the object to operate under default control. In this case, the values assigned to Default Current, Default Next, and Default Time will be placed at the outputs, Current, Next, and Time, respectively. In addition, the Active Event output will be set to a not active (NA). The Control Status output value reflects the output source by indicating default conditions (Control Status = 0).	—	Disabling the 7-Day Schedule object causes the algorithm to clear all assigned temporary override values. This also provides the user a method for manually clearing temporary overrides if required.
Excep1	Exception 1	Class: Digital — An exception schedule is activated when the corresponding Exception input is set to a Digital ON condition. An input of Digital ON at Exception 1, 2, 3, or 4 causes the algorithm to utilize the particular Exception schedule in place of the regular schedule. The exception schedule will continue to be used as long as the particular Exception input remains ON. An Exception input of Digital OFF or not active (NA) causes the object to disregard the particular Exception.	—	Simultaneous activation of multiple exceptions causes the algorithm to use the exception with the highest priority. The object will evaluate the exception inputs using a high Exception 1 to low Exception 4 priority.
Excep2	Exception 2			
Excep3	Exception 3			
Excep4	Exception 4			
FrcOvrd	Force Override	Class: Digital — A Forced Override input of Digital ON causes values assigned to the inputs Override Current, Override Next, and Override Time to be placed at the outputs, Current, Next, and Time, respectively. In addition, the Active Event output will be set to a not active (NA). The Control Status output value reflects the output source by indicating a forced override condition (Control Status = 1). A Forced Override input of Digital OFF or not active (NA) causes the object to disregard the Force Override function.	—	An active Force Override will always output the assigned values regardless of the validity of the actual referenced time, the temporary override conditions, the conditions of the exception inputs, and the active regular schedule status.
OvrdCrnt	Override Current	Class: Analog — A Forced Override input of Digital ON causes the value assigned to Override Current to be placed at the Current output.	Based upon the Event Format selected. For value definitions, refer to Notes given for the Event Format property.	
OvrdNext	Override Next	Class: Analog — A Forced Override input of Digital ON causes the value assigned to Override Next to be placed at the Next output.		
OvrdTime	Override Time	Class: Analog — A Forced Override input of Digital ON causes the value assigned to Override Time to be placed at the Time output.	-163.83 to 16383 and (NA)	

Table–5.263 Schedule 7-Day Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Current	Current	Class: Analog — Reflects the currently active event value. The algorithm determines the active event value by analyzing the enable, force override, temporary override, exception, and regular schedule functions and sets the Current output accordingly.	Based upon the Event Format selected. For value definitions, refer to Notes given for the Event Format property.
Next	Next	Class: Analog — Reflects the next scheduled event value. The algorithm determines the next event value by analyzing the enable, force override, temporary override, exception, and regular schedule functions and sets the Next output accordingly.	
Time	Time	Class: Analog — The Time output reflects the calculated number of minutes until the next event value becomes the active event value.	-163.83 to 16383 and (NA)
ActEvnt	Active Event	Class: Analog — The Active Event value reflects the temporary, exception, or regular scheduled event in process by indicating an enumerated value between 0 and 4 or NA.	0 – The event is a continuation of an active event from a previous day(s) 1 – Event 1 Active 2 – Event 2 Active 3 – Event 3 Active 4 – Event 4 Active NA – Scheduled disabled, Force Override active, or controller time not valid
Status	Control Status	Class: Analog The Control Status value reflects the output source for the Current, Next, and Time outputs by indicating an enumerated value between 0 and 7.	0 – Default Conditions Active 1 – Forced Override Active 2 – Temporary Schedule Active 3 – Exception 1 Active 4 – Exception 2 Active 5 – Exception 3 Active 6 – Exception 4 Active 7 – Regular Schedule Active

Applying the Object

The 7-Day Schedule object provides a means for scheduling a seven day, repeating set of events. Each day can be programmed to provide up to four scheduled events. In addition, the 7-Day Schedule object provides exception handling for up to four exception conditions. The 7-Day Schedule object works in conjunction with the Calendar object to provide a complete solution for yearly schedule control.

Schedule Enable/Disable

The 7-Day Schedule function is enabled by a Schedule Enable input of Digital ON or not active (NA). A Schedule Enable input of Digital OFF disables the 7-Day Schedule, causing the object to operate under default control. In such a case, the values assigned to Default Current, Default Next, and Default Time will be placed at the Current, Next, and Time outputs, respectively. In addition, the Active Event output will be set to not active (NA). The Control Status value reflects the output source by indicating default conditions (Control Status = 0). Disabling the object causes the algorithm to clear all assigned temporary override values. This may also serve as a method for manually clearing temporary overrides if required.

In a typical application, the value assigned to the Default Current configuration parameter reflects the control fallback to be used during object disable or invalid controller time conditions. The values assigned to the Default Next and Default Time configuration parameters are typically set to not active (NA), to allow controlled transitions when interfaced to advanced objects such as Optimum Start Stop. These advanced objects are designed to specifically handle not active (NA) conditions. However, if a value other than not active (NA) is assigned to the Default Time configuration parameter, it will be placed at the Time output and will not decrement as is typically seen when schedules are active.

Forced Override

The Forced Override function allows the direct override of regular schedules, exceptions, and temporary scheduled events. This is done by using the Forced Override, Override Current, Override Next, and Override Time inputs. The forced override function allows a schedule from a remote source, such as a network variable input object or cascaded 7-Day Schedule object, to be utilized regardless of local object conditions. The forced override function is ignored whenever the object is disabled.

A Forced Override input of Digital ON causes the values assigned to the inputs Override Current, Override Next, and Override Time to be placed at the outputs Current, Next, and Time, respectively. In addition, the Active Event output is set to a not active (NA). The Control Status value reflects the output source by indicating a forced override condition (Control Status = 1). An active Force Override will always output the assigned values regardless of the validity of the actual referenced time, the temporary override conditions, the conditions of the exception inputs, or the active regular schedule status. A Forced Override input of Digital OFF or not active (NA) causes the object to disregard the Force Override function.

Valid/Invalid Time

The 7-Day Schedule object uses the time that is internally maintained by the controller platform and will process the schedule as long as the time / day parameters are determined to be valid. The controller clock typically becomes valid once it is synchronized through a local or remote clock source.

An invalid controller time causes the 7-Day Schedule object to operate under default conditions. When this occurs, the values assigned to Default Current, Default Next, and Default Time are placed at the outputs Current, Next, and Time, respectively, as long as a force override condition is not in process. In addition, the Active Event output is set to not active (NA). The Control Status value reflects the output source by indicating default conditions (Control Status = 0). Invalid time will also cause the algorithm to clear all assigned temporary override values.

Schedules

The 7-Day Schedule object supports the programming of a seven day regular schedule, a seven day temporary schedule, and one to four exception schedules, which are used by the algorithm during active exception input conditions.

Regular Schedules

Regular schedules consist of seven days, with each day allowing the assignment of one to four scheduled events. Regular schedules are programmed using the daily schedule configuration properties (Sunday Schedule [1-4], Sunday Event [1-4], Monday Schedule [1-4], Monday Event [1-4], etc.). The events within each day are programmed by assigning a valid event time with a corresponding event value. Regular schedule times and events are programmed and stored in nonvolatile memory.

Schedule time values are defined in hours and minutes (HH:MM). Hours (HH) range from 00 to 23. Minutes (MM) range from 00 to 59. A schedule time is considered invalid whenever the assigned time is set to a not active (NA) or to any time which is outside the predefined hour / minute range. An invalid schedule time causes the particular schedule event to be bypassed.

Daily scheduled times must be programmed in ascending chronological order for proper operation. An invalid schedule time or a time value that is out of sequence will cause the particular schedule event to be bypassed or ignored. The schedule object will run at least the first valid schedule time within the active day.

Temporary Schedules

Note: While temporary schedules are a part of the 7-Day Schedule object, they are not implemented with WP Tech. A description of this feature is provided here for information only. The temporary schedule feature is reserved for use with future Human Machine Interface (HMI) products.

The 7-Day Schedule object includes a temporary schedule option which provides a means for temporarily modifying a regular schedule, up to one week in advance. A valid temporarily scheduled event will act as a replacement for the regularly scheduled event, and will be used by the algorithm for calculation of schedule output conditions.

Exceptions

The 7-Day Schedule object provides up to four exception schedules, used by the algorithm during active exception input conditions.

Each exception schedule allows the assignment of up to four scheduled events. The exception schedules are programmed using the exception schedule configuration properties (Exception 1 Schedule [1-4], Exception 1 Event [1-4], Exception 2 Schedule [1-4], Exception 2 Event [1-4], etc.). The four events within each exception schedule are programmed through the assignment of a valid event time with a corresponding event value. Exception schedule times and events are programmed and stored in nonvolatile memory.

Exception schedule time values are defined in hours and minutes (HH:MM). Hours (HH) range from 00 to 23, with Minutes ranging from 00 to 59. A schedule time is considered invalid whenever the assigned time is set to a not active (NA), or to any time which is out of the predefined hour / minute range. An invalid schedule time causes the particular schedule event to be bypassed.

Exception scheduled times must be programmed in ascending chronological order for proper operation. An invalid schedule time or a time value that is out of sequence will cause the particular schedule event to be bypassed or ignored. The schedule object will run at least the first valid schedule time within the active exception schedule.

Exception schedule event values must follow the same format used to program the regular schedule events. This format is defined by the Event Format configuration property.

An exception schedule will be activated when the corresponding Exception input is set to a Digital ON condition. An Exception input of Digital ON causes the algorithm to utilize the corresponding exception schedule in place of the regular schedule. The exception schedule will continue to be used as long as the Exception input remains ON. Simultaneous activation of multiple exceptions causes the algorithm to use the exception with the highest priority. The object will evaluate the exception inputs, prioritizing them high (Exception 1) to low (Exception 4). An Exception input of Digital OFF or not active (NA) causes the object to disregard the exception condition.

Schedule Operation

The 7-Day Schedule object's algorithm dynamically calculates all outputs based upon control conditions, including schedule enable, forced override, temporary overrides, exceptions, and regular schedule assignments. The algorithm continuously reviews input / schedule conditions and calculates the output conditions based upon a predefined process order.

A 7-Day Schedule object that is not enabled causes the object to operate under default control. In such a case, the values assigned to Default Current, Default Next, and Default Time are placed at the outputs Current, Next, and Time, respectively. In addition, the Active Event output will be set to a not active (NA). The Control Status value reflects the output source by indicating default conditions (Control Status = 0). Disabling the object causes the algorithm to clear all assigned temporary override values.

When the object is enabled, the algorithm monitors the Force Override input to determine if a force condition exists. A Forced Override input of Digital ON causes values assigned to the inputs Override Current, Override Next, and Override Time to be placed at the outputs Current, Next, and Time, respectively. In addition, the Active Event output is set to a not active (NA). The Control Status value reflects the output source by indicating a forced override condition (Control Status = 1).

In all cases, the 7-Day Schedule object continuously compares the controller's reference time to the programmed schedule times and dynamically maintains the current status of internal scheduled events. The algorithm continues to calculate programmed regular, temporary, and exception schedules regardless of schedule enable and force override conditions. This allows a smooth transition when returning to the scheduled program when the object is enabled, or when the force override condition is removed. Operating the schedules in the background allows temporary schedules to be cleared as required.

The object's algorithm determines the current status of internal scheduled events, processing schedules in order, with temporary schedule first, exception schedule second, and regular schedule third.

A temporary scheduled event has the highest priority. When activated, the Active Event output will be set to indicate the event in process (0 through 4). The Control Status value reflects the output source by indicating a temporary schedule condition (Control Status = 2), and the Current, Next, and Time outputs are adjusted accordingly. Activation and completion of a temporary scheduled event causes the algorithm to replace the temporary scheduled event with the corresponding regular schedule event, retrieved from nonvolatile memory. In the case where multiple temporary schedule events are completed, the algorithm replaces them with the corresponding regular schedule events, retrieved from nonvolatile memory.

When, under active temporary schedule conditions, adjustments to the controller's time cause the referenced time to advance or setback more than one minute, the algorithm completes the temporary schedule as follows:

- A time that advances past the next scheduled event(s) causes the algorithm to retrieve all the corresponding regular schedule events from nonvolatile memory, including the schedule where the temporary schedule was initiated.
- A time that sets back to when the temporary scheduled event began causes the algorithm to recalculate and proceed forward with the schedule object operation.

Whenever a temporary schedule is not active, the algorithm processes an active exception schedule. An active exception condition causes the algorithm to utilize the corresponding exception schedule in place of the regular schedule. The exception schedule continues to be used as long as the associated Exception input remains ON.

Simultaneous activation of multiple exceptions causes the algorithm to use the exception with the highest priority. The object will evaluate the exception inputs, prioritizing them high (Exception 1) to low (Exception 4). During an exception schedule condition, the Active Event output is set to indicate the exception event in process (0, 1, 2, 3, or 4). The Control Status value reflects the output source by indicating an exception condition of Exception 1, Exception 2, Exception 3, or Exception 4 (Control Status = 3, 4, 5, or 6, respectively), and the Current, Next, and Time outputs are adjusted accordingly.

The algorithm processes the regular schedule if either temporary or exception conditions are not active. Under these conditions, the Active Event output will be set to indicate the event in process (0, 1, 2, 3, or 4). The Control Status value reflects the output source by indicating a regular schedule condition (Control Status = 7), and the Current, Next, and Time outputs are adjusted accordingly.

In the case of a controller reset or power cycle, the temporary schedules are cleared and the algorithm proceeds with schedule object operation.

Outputs

The outputs of the 7-Day Schedule object include Current, Next, Time, Active Event, and Control Status.

Current

The Current output reflects the currently active event value. The algorithm determines this value by analyzing the enable, force override, temporary override, exception, and regular schedule functions, and sets the Current output accordingly.

Next

The Next output reflects the next scheduled event value. The algorithm determines this value by analyzing the enable, force override, temporary override, exception, and regular schedule functions, and sets the Next output accordingly.

Time

The Time output reflects the calculated number of minutes until the next event value becomes the active event value.

Active Event

The Active Event output indicates the schedule event in process. Each temporary, exception, and regular schedule allows programming of up to four events per day. The Active Event value ranges from 0 to 4. An Active Event value of 1, 2, 3, or 4 directly relates to the corresponding programmed event within the schedule.

An Active Event value of 0 (zero) indicates that the event is a continuation of an active event from a previous day(s). The Active Event will indicate a not active (NA) whenever the 7-Day Schedule object is disabled. A not active (NA) will also occur whenever Force Override is activated or the controller time referenced by the object is found to be invalid.

Control Status

The Control Status value reflects the output source for the Current, Next, and Time outputs, by indicating a value between 0 and 7. Control Status is an enumerated value defined as shown in Table–5.264.

Table–5.264 Control Status Enumeration.

Control Status	Description
0	Outputs reflect the assigned default configuration values.
1	Outputs reflect the values assigned to the Force Override inputs.
2	Indicates a temporary schedule in process.
3	Indicates Exception 1 active.
4	Indicates Exception 2 active.
5	Indicates Exception 3 active.
6	Indicates Exception 4 active.
7	Indicates a regular schedule in process.

Example

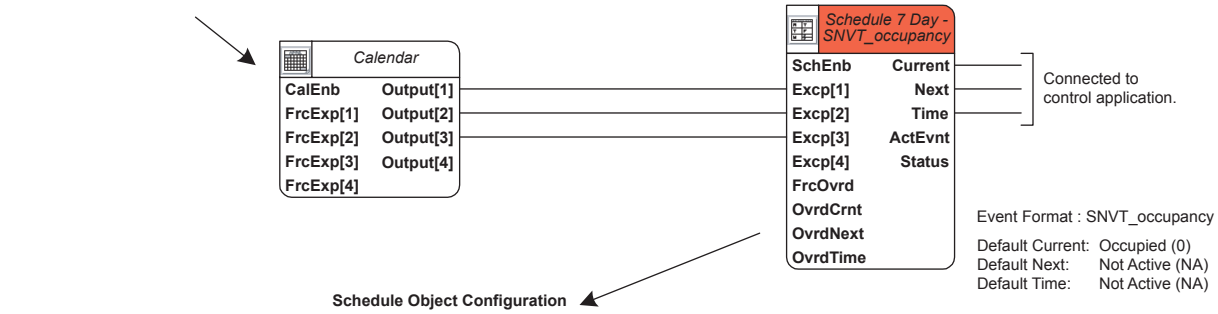
The following example illustrates the use of a 7-Day Schedule object, in conjunction with a Calendar object, to provide yearly schedule operation.

A 7-Day Schedule object is programmed to provide a weekly schedule in which the building is occupied between the hours of 7 a.m. and 5 p.m., Monday through Friday, as well as the time between the hours of 7 a.m. and 12:30 p.m. on Saturday. See Figure–5.146.

Three exception schedules have also been programmed, to provide alternate schedules whenever an exception event (holiday, etc.) occurs. In this example, Exception 1 causes the schedule output to remain unoccupied on New Years Day, Memorial Day, Independence Day, Labor Day, Thanksgiving, and Christmas. Exception 2 causes the schedule output to extend the occupied hours for Thursday Board Meetings, which occur on the second Thursday of every month. Exception 3 causes the schedule output to return to occupied mode for a Scout meeting, which occurs every Tuesday evening from September through May.

Calendar Object Configuration

	Program Type	Start Date	Start Time	End Date	End Time	Day Select	Duration	Output Select
New Years Day	Event	01/01	00:00	01/01	23:59	---	---	1
Memorial Day	Last Occur	05/01	00:00	05/31	---	Monday	24.0	1
Independence Day	Event	07/04	00:00	07/04	23:59	---	---	1
Labor Day	1st Occur	09/01	00:00	09/30	---	Monday	24.0	1
Thanksgiving	4th Occur	11/01	00:00	11/31	---	Thursday	24.0	1
Christmas	Event	12/25	00:00	12/25	23:59	---	---	1
Board Meeting	2nd Occur	01/01	00:00	12/31	---	Thursday	24.0	2
Scout Meeting	All Occur	09/01	18:30	05/31	---	Tuesday	3.0	3



Schedule Object Configuration

		Sun	Mon	Tue	Wed	Thu	Fri	Sat
Event 1	Time	---	07:00	07:00	07:00	07:00	07:00	07:00
	Event	---	Occ (0)	Occ (0)	Occ (0)	Occ (0)	Occ (0)	Occ (0)
Event 2	Time	---	17:00	17:00	17:00	17:00	17:00	12:30
	Event	---	Unocc (1)	Unocc (1)	Unocc (1)	Unocc (1)	Unocc (1)	Unocc (1)
Event 3	Time	---	---	---	---	---	---	---
	Event	---	---	---	---	---	---	---
Event 4	Time	---	---	---	---	---	---	---
	Event	---	---	---	---	---	---	---

		Exception 1	Exception 2	Exception 3	Exception 4
Event 1	Time	00:00	07:00	18:30	---
	Event	Unocc (1)	Occ (0)	Occ (0)	---
Event 2	Time	---	22:00	21:30	---
	Event	---	Unocc (1)	Unocc (1)	---
Event 3	Time	---	---	---	---
	Event	---	---	---	---
Event 4	Time	---	---	---	---
	Event	---	---	---	---

Figure–5.146 Example of Schedule 7-Day Object Application.

Select

Object Usage: The Select object passes one of the values at either Input[1] or Input[2] as the Output value, based on the current state (ON or OFF) at the Input Select input. In effect, the Select object acts as a single-pole, double-throw switch, selecting between the two inputs. Values connected to Inputs[1] and [2] are typically analog types, but may be digital types if required.

Device Support: (all firmware revisions)

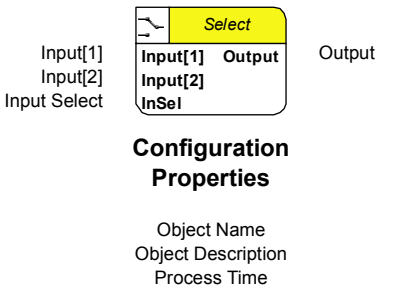
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.265 Select Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.266 Select Object Input Properties

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input[1]	Class: Analog -The first of two inputs selectable as the Output. This value appears at the Output if an OFF or not active (NA) is at the Input Select input.	-163.83 to 16383	

Table–5.266 Select Object Input Properties

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[2]	Input[2]	Class: Analog - The second of two inputs that can be selected as the Output. This value appears at the Output if an ON is at the Input Select input.	-163.83 to 16383	
InSel	Input Select	Class: Digital - Used to switch selection between the values at Input[1] and Input[2].	—	If not active (NA), the Output value is Input[1].

Table–5.267 Select Object Output Properties

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - Reflects the value at either Input[1] or Input[2], based on the digital state at the Input Select input. A not active (NA) indicates the selected input (Input[1] or [2]) is NA.	-163.83 to 16383

Applying the Object

The Select object provides a means to choose from one of two values based on the current digital state at the Input Select input.

- Input[1] is the OFF selection, meaning it appears at the object Output with either an OFF state or not active (NA) at the Input Select input.
- Input[2] is the active or ON selection, meaning it appears at the object Output only if an ON state is present at the Input Select input.

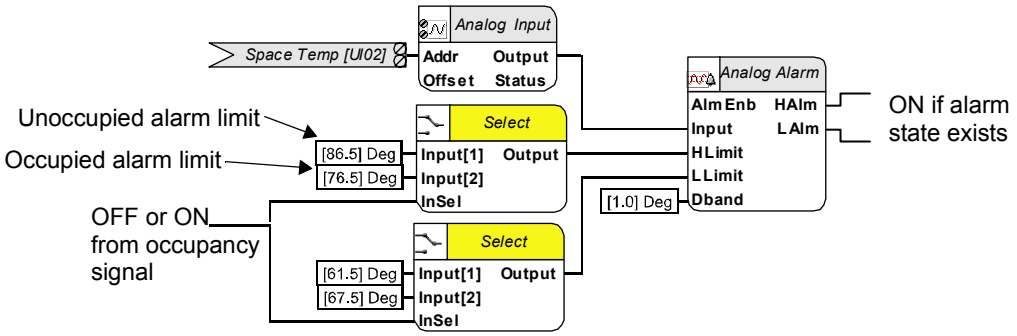
Table–5.268 shows all possible input to output combinations, including NAs.

Table–5.268 Truth Table for Select Object.

Inputs			Output
Input Select	Input[1]	Input[2]	
OFF	AV1	AV2	AV1
OFF	AV1	NA	AV1
OFF	NA	AV2	NA
ON	AV1	AV2	AV2
ON	AV1	NA	NA
ON	NA	AV2	AV2
NA	AV1	AV2	AV1
NA	AV1	NA	AV1
NA	NA	AV2	NA

Control Logic Example

Two Select objects in this example Figure-5.147 are used to establish different high and low alarm limits for an Analog Alarm object that provides alarm detection for a space temperature value.



Figure–5.147 Example Select Object Control Application.

Each Select object has its Input Select input connected to a digital value corresponding to a schedule OFF/ON state. When this digital value is OFF, the Output of each Select object represents the unoccupied alarm limit at Input[1]. When this digital value is ON, the Output of each Select object represents the occupied alarm limit at Input[2].

Sensor Input

Object Usage: The Sensor Input object supports the means for interfacing the application to analog-sensor-type physical input hardware. This point-type object is similar to the Analog Input object, supporting the same base set of Resistive Type Devices (RTDs) with the addition of the copper style RTDs. In addition, the sensor input object can be configured to support a temperature sensor equipped with a “shorting-type” pushbutton switch, such as the TS-90XXX series. The shorting of the sensor can be used to start or cancel a timed override function. The sensor input object monitors the assigned hardware input and determines the proper object output value, based upon the selected sensor type.

Input sensor selections include:

- Thermistor (10k with 11k shunt)
- Balco
- Platinum
- Copper

Each Sensor Input object requires one UI input on an MN 800.

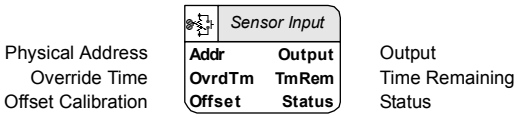
Device Support:

MN 800 series

Memory Requirements: (per object)

EEPROM: 16 bytes
RAM: 14 bytes

WP Tech Representation



Configuration Properties

Object Name
Object Description
Input Sensor Type
Switch Type
Filter Constant

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.269 Sensor Input Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
Type	Input Sensor Type	Class: Analog - This value defines the sensor type connected to the hardware. Note: Only RTD-type sensors are supported.	1	1 - 10k Thermistor 2 - Balco 3 - Platinum 9 - Copper	A 10k Thermistor RTD must have an 11k shunt, such as with any of the TS-87xx-850 series.

Table–5.269 Sensor Input Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selections	Notes
Switch	Switch Type	Class: Analog - This value defines the selections available for the shorting switch (used for the associated override) and timed control. The selections are as follow: 0 - No Switch - No overrides possible from using the sensor shorting switch. 1 - Pushbutton Override without Cancel - The override is restarted each time the sensor is shorted by the switch. It cannot be canceled by the switch. 2 - Pushbutton Override with Cancel - The override can be started, restarted, and canceled by the shorting switch. See the Notes at right. 3 - Continuous Switch - The override is activated each time the sensor is shorted by the switch, and canceled when the switch is open. Whenever the override is activated, the measured temperature value is held (frozen) as long as the switch is closed, and diagnostic errors are not indicated at the Status Flags output. ***Note:*** For any Input Switch Type setting except “Continuous”, a diagnostic error is reported whenever the sensor is shorted > 9 seconds, whereby the object Output goes to not active (NA) and the Status Flags output indicates an error condition.	0	0 - No Switch 1 - Pushbutton Override without Cancel 2 - Pushbutton Override with Cancel 3 - Continuous Switch	Pushbutton with Cancel (2) is the typical selection. This provides the same override function available with an MN-S2 or higher sensor. In this configuration, the shorting switch pushbutton works as follows: • Pressing < 4 seconds *initiates* an override. If already active, it is *restarted* to its full duration. • Pressing and holding > 4 but < 9 seconds *cancels* an active override. • Pressing and holding > 9 seconds results in *diagnostic errors*.
Filter	Filter Constant	Class: Analog -Defines the filter constant or filter factor applied to the input. Maximum Filter is at 0.01 Minimum Filter is at 0.99	1	0.00 to 1.00	An not active (NA), 0.00, or 1.00 bypasses the filtering action.

Table–5.270 Sensor Input Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Addr	Physical Address	Class: Analog - Indicates the physical hardware address (input terminal point on the controller) assigned to the Sensor Input object.	Dependent on the controller platform selected.	If no physical hardware address is assigned (not active or NA), outputs are also set to not active (NA).

Table–5.270 Sensor Input Object Input Properties. (Continued)

Abbrev.	Name	Class / Description	Range / Selections	Notes
OvrdTm	Override Time	Class: Analog - Determines the length of the timed override periods, in minutes, initiated from the sensor's shorting switch. The input value is typically a value from 1 to 16382 and is usually sourced from a constant value or NCI tag. • A negative value, 0, or not active (NA) acts as zero (no override time). In this case, no override occurs (if an override is active it is canceled). Shorting the switch has no effect. • A value of 16383 causes a "forever" override where the override timer does *not* decrement. The cancelling of an active "forever" override is dependent on the Switch Type assigned (see Applying the Object, Switch Type description).	0 to 16383	***Note:*** During an active override, any changes to the Override Time value is processed immediately. For example, if 5 minutes have already passed during an active override, and the Override Time value is changed from 10 minutes to 20 minutes, the output would immediately change to 15 minutes remaining. A 0 (zero) value cancels the active override and prevents subsequent sensor overrides. A value of 16383 causes a "forever" override condition.
Offset	Offset Calibration	Class: Analog - Defines the value added to or subtracted from the calculated output value prior to placing the value at the actual output.	-163.83 to 16383	If unconnected or with a NA value, the Offset Calibration defaults to 0.0 (no offset applied).

Table–5.271 Sensor Input Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The calculated Sensor Input object output value. This output will indicate not active (NA) whenever the Sensor Input object is not assigned a valid physical address, or when the data from the actual assigned hardware point is not valid (for example, the sensor is out-of-range for the selected sensor type). The output returns to the normal condition when the sensor signal is in the proper range. ***Note:*** The output value is held while the sensor shorting switch is closed. If the shorting switch is closed longer than 9 seconds, the output goes to NA (unless the Input Switch Type is set to Continuous).	-163.83 to 16383
TmRem	Time Remaining	Class: Analog - Indicates the remaining time, in minutes, in a timed override sequence initiated by the sensor's shorting switch. The duration of the override period is defined by the value at the Override Time input. After an override is initiated, the value counts down each minute towards 0 (zero), at which time the override expires (see Note below). After an override expires or an override is canceled by the sensor's shorting switch, the output remains at 0 (zero). ***Note:*** A value of 16383 means an active "forever" override, which does NOT count down. The Time Remaining output can also be used digitally as follows: 0 = Digital OFF >0 = Digital ON, indicating the active override condition	0 (No Override) or 1 to 16383 (Active Override)

Table–5.271 Sensor Input Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Status	Status	Class: Analog - The Status output is an enumerated value that represents current error conditions within the Sensor Input object, as follows: 0 - No errors; normal object operation. 1 - Physical Address not active (NA) or invalid. 2 - Sensor under-range condition. 3 - Sensor over-range condition. This output is set to an error value whenever an error condition is determined by the Sensor Input algorithm. An error condition exists whenever the Sensor Input object is not assigned a valid physical address or the data from the actual assigned hardware point is not valid (signal out-of-range condition). The output returns to normal (0) when the input signal is within the normal range. ***Notes:*** • The Status output does not change value if the sensor's shorting switch is closed for less than 9 seconds. If the shorting switch is closed longer than 9 seconds, the Status output indicates either 2 or 3, based upon the sensor type, and the Output goes to NA (unless the Input Switch Type is set to Continuous). • This output will never reflect a sensor under-range (over-range for thermistor) condition error if the Input Switch Type is set to Continuous.	0 (Normal) 1, 2, or 3 (Active Error)

Applying the Object

The Sensor Input object allows the MN 800 to support a temperature sensor equipped with a "shorting-type" pushbutton switch, such as the TS-90XXX series. This control scheme is useful because it provides both a temperature reading and an override-to-occupancy signal from a single wire pair, UI controller input, and low-cost, wall mounted, RTD room sensor.

The Sensor Input object combines the functions of an Analog Input object used for an RTD sensor with the override capability of an MN-S2 or higher sensor. The object has one input and two outputs for the override feature; other inputs and outputs are like those on the Analog Input object.

Input Sensor Type

The Sensor Input object can be assigned to monitor any of the physical universal input (UI) points on the I/A Series MicroNet controller. The object's Input Sensor Type assignment configures the hardware terminals to support any of the following standard RTD sensor types:

- Thermistor (10k with 11k shunt)
- Balco
- Platinum
- Copper

All four selections cause the Sensor Input object to calculate the output values using controller-resident characterization, provided by the hardware. Offset Calibration as well as the Filter function may be applied to the output value of the object. Typical sensors for the four selections are listed in Table–5.272.

Table–5.272 Typical Sensors Interfaced with Sensor Input Object.

Input Sensor Type	Typical Sensors	
	Model Number	Description
Thermistor (10k with 11k Shunt)	TS-5711-850 Series TS-570X1-850 Series	Room Thermistor Temperature Sensors (10k thermistor sensors with 11k shunt resistor)
	TSMN-57011-850	Electronic Room Temperature Sensor (contemporary wall sensor)
	TS-90XXX Series[a]	MicroZone II-Style 10k Thermistor Wall Sensors with 11k Shunt Resistor
Balco	TS-8000 Series	Remote Balco Temperature Sensors
	TSMN-81011	Electronic Room Temperature Sensor (contemporary wall sensor)
Platinum	TS-5800 series	Remote Platinum Temperature Sensors
	TSMN-58011	Electronic Room Temperature Sensor (contemporary wall sensor)
Copper	TS-5900 Series	Copper Sensors

a. The TS-90XXX series sensors include a pushbutton to momentarily short the RTD. These sensors can perform a timed override function by incorporating the object's pushbutton override feature.

Switch Type

The Switch Type assignment defines the selections available for pushbutton switch / timed override control. These selections include:

- No Switch
- Pushbutton Override without Cancel
- Pushbutton Override with Cancel
- Continuous Switch

The Sensor Input object performs the Switch Type sequence whenever the algorithm detects a shorted sensor condition (i.e. input resistance is less than 300 ohms).

No Switch

A Switch Type of "No Switch" disables the use of the timed override function, causing the Time Remaining output to remain not active (NA). A shorted sensor causes the Sensor Input object to interpret the value to be outside the valid sensor range, forcing the Status output to indicate a sensor out-of-range condition with all the other outputs being set accordingly.

Pushbutton Override without Cancel

A Switch Type of "Pushbutton Override without Cancel" enables the use of a momentarily shorted sensor to initiate a timed override. In this case, an override is initiated by shorting the sensor (pressing and holding the TS-90XXX Series pushbutton) for less than 9 seconds, typically between 1 and 9 seconds. During this time period, the Sensor Input object's output is held at the last calculated value.

If the sensor is shorted for more than 9 seconds, this is considered a true “short”, causing the Status output to indicate a sensor out-of-range condition, with all the other outputs being set accordingly. See the Status output diagnostic table Table–5.274 for further details.

An active override causes the Sensor Input algorithm to set the Time Remaining output to the value assigned to the Override Time input and, once the short is removed, decrement the Time Remaining output until it is returned to a 0 (zero) value. The override is restarted each time the switch is used to short the sensor. An assigned Override Time of 0 (zero) cancels an active override and prevents subsequent overrides as long as this value is 0. An assigned Override Time of 16383 causes a “forever” override condition when the override is active. This override condition is canceled when the Override Time is changed to 0 (zero) or the controller is reset.

Pushbutton Override with Cancel

A Switch Type of “Pushbutton Override with Cancel” enables the use of a momentarily shorted sensor for both initiating and canceling a timed override. In this case, shorting of the sensor (pressing and holding the TS-90XXX Series pushbutton) for less than 4 seconds (typically between 1 and 4 seconds) initiates an override. Shorting of the sensor for more than 4 seconds, but less than 9 seconds, cancels an active override. During this time period, the Sensor Input object’s output will be held at the last calculated value. Shorting of the sensor for more than 9 seconds is considered a true “short”, causing the Status output to indicate a sensor out-of-range condition, with all the other outputs being set accordingly. The override is restarted each time the switch is used to short the sensor. See the Status output diagnostic table Table–5.274 for further details.

An active override causes the Sensor Input algorithm to set the Time Remaining output to the value assigned to the Override Time input and, once the short is removed, decrement the Time Remaining output until it is returned to a 0 (zero) value. An assigned Override Time of 0 (zero) cancels an active override and prevents subsequent overrides as long as this value is 0. An assigned Override Time of 16383 causes a “forever” override condition when the override is active. This override condition is canceled when the sensor is shorted for more than 4 seconds but less than 9 seconds, when the Override Time is changed to 0 (zero), or when the controller is reset.

Continuous Switch

A Switch Type of “Continuous Switch” enables the use of a continuously shorted sensor to perform the continuous switch function.

For example, a two-position switch (i.e. low limit thermostat) is wired across the sensor. Continuous shorting of this sensor for more than 9 seconds causes the Sensor Input algorithm to hold the object’s output to the last calculated value and set the Time Remaining output to the value assigned to the Override Time input. In this case, the algorithm will not decrement the Time Remaining output while the short is maintained. Once the short is removed, the Output will be released for control and the Time Remaining output will be returned to a 0 (zero) value. During “Continuous Switch” operation, a shorted sensor condition will never be reflected as an error at the Status output. Open sensors will still generate an error at the status output.

Filter Constant

A Filter Constant can be applied to perform an exponential low pass filter which limits the response of the output in relationship to a step change at the input. A filter is typically used to dampen the effects of input change to output change, to stabilize a noisy or rapidly changing input signal. The value of the Filter Constant property is limited to values between 0.00 and 1.00. Filter Constant action is shown in Table–5.273 below.

Table–5.273 Filter Constant and Filter Action.

Filter Constant	Filter Action
0.00	No Filter
0.01 (Maximum Filter) through 0.99 (Minimum Filter)	Active Filter Area
1.00 (Default)	No Filter
Not Active (NA)	No Filter

Between any two successive changes at the input, the filter algorithm provides the function as follows:

Output = Prev. Output + [Filter Constant X (Present Input - Prev. Output)]

For example, a Sensor Input object for an airflow sensor is observed to have an unstable output near the sensor's low input range, with the object output constantly jumping between 50 and 75 when airflow is holding near 60 CFM. By using a Filter Constant of 0.30, this jumping is minimized:

At first Input jumps from 50 to 75:

Output = 50.00 + [0.30 (75.0 - 50.00)] or **57.50** (vs. 75.0 with No Filter)

Input jumps back to 50:

Output = 57.50 + [0.30 (50.0 - 57.50)] or **55.25** (vs. 50.0 with No Filter)

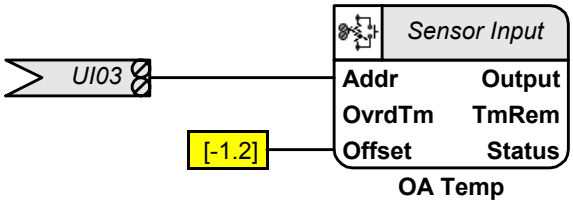
Input jumps up to 75:

Output = 55.25 + [0.30 (75.0 - 55.25)] or **61.18** (vs. 75.0, and so on)

Offset Calibration

Offset calibration defines a value that is added to the calculated output value prior to placing the value at the actual object output. A positive offset value increases the value of the output; a negative offset value decreases the value of the output.

Offset calibration is an input of the Sensor Input object. If used, a constant value is typically assigned (connected) as the offset. An unconnected Offset input is equivalent to a not active (NA), causing the Offset Calibration to be set to 0.0 (no offset applied). Offset is useful in calibrating a sensor to a known accurate measuring device, such as a temperature sensor calibrated to a precision thermometer. In the example Sensor Input object below Figure-5.148, a -1.2 constant was used to calibrate the sensor reading from a value that (with no offset) was measured to be 1.2 degrees too high.



Figure–5.148 Example Sensor Input Object Using Offset Calibration.

Status Output

The Status output is 0 (zero) under normal conditions. This output is set to an enumerated value whenever the Sensor Input algorithm detects an error condition. See Table–5.274, below, for Status output values.

Table–5.274 Sensor Input Object Status Output Error Codes.

Diagnostic Condition	Status	Output and Time Remaining
No errors; normal object operation.	0	Normal Operation NA if Type = Disabled
Physical Address not active (NA) or invalid.	1	Set to not active (NA)
Sensor under-range condition.	2	Set to not active (NA)
Sensor over-range condition.	3	Set to not active (NA)

Note:

- For any Switch Type setting except “Continuous Switch,” a diagnostic error will be generated whenever the sensor is shorted for more than 9 seconds. For this diagnostic condition, Balco, Platinum, and Copper Input Sensor Types will indicate a sensor under-range Status value (2). The 10k Thermistor Input Sensor Type will indicate a sensor over-range Status value (3) due to the sensor’s negative temperature coefficient properties. For all Switch Type settings, an open or high out-of-range resistance value will generate an error condition. With BALCO, platinum, or copper sensors, this diagnostic condition is an over-range condition. With 10k thermistor sensors, this diagnostic condition is an under-range condition.
- Upon reset or return from an error condition (return from status error to normal operation), the timed override function will be initialized to the “no override” condition. The Time Remaining output will be initialized to a value of 0 (zero).
- The Status output can also be used digitally as a value of zero reflects digital OFF and a value greater than zero reflects a digital ON, indicating an error condition.

Error conditions include sensor under-range, sensor over-range, and improper setup (no valid physical address). Refer to Table–5.275 for the operating and diagnostic trip points and ranges for the various RTD sensor types when used with I/A Series MicroNet controllers.

Note: The values given in Table–5.275 are general operating ranges. Refer to the RTD specification data sheets for details.

Table–5.275 Operating and Diagnostic Trip Points and Ranges for RTD Sensors.

RTD Type	Operating Range	Valid to NA		Return from NA	
		Low	High	Low	High
Thermistor Balco	-40 to 250 °F (-40 to 121 °C)	Approx. -49 °F (-45 °C)	Approx. 267 °F (131 °C)	Approx. -41 °F (-40.5 °C)	Approx. 251 °F (122 °C)
Platinum Copper	-40 to 240 °F (-40 to 116 °C)	Approx. -44 °F (-42 °C)	Approx. 256 °F (124 °C)	Approx. -41 °F (-40.5 °C)	Approx. 241 °F (116 °C)

Sequence (3)

Object Usage: The Sequence (3) object provides a method to sequence outputs based on the proportional 0.0 to 100.0% signal received at the object Input. Outputs[1] to [3] can function as either binary outputs (OFF or ON) or as analog outputs (0.0 to 100.0%), or as a combination of the two, depending on the Sequence Mode setting. The Sequence Mode can be set to one of the following:

- Linear Sequence Mode - outputs operate in a binary (OFF or ON), first-on / last-off logical sequence.
- Analog Sequence Mode - outputs operate in an analog (0.0 to 100.0%) continuously proportional sequence.
- Vernier Sequence Mode - a single output operates in a proportionally analog fashion, while the remaining two outputs offer additional binary (OFF or ON) sequence control.

The Sequence (3) object operates like the Sequence (6) (page 481) and Sequence (10) (page 488) objects, but with fewer total outputs.

Device Support: (all firmware revisions)

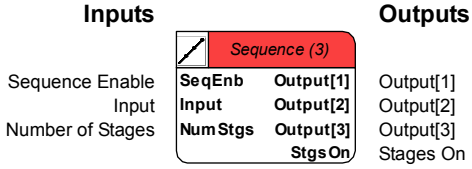
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 12 bytes
RAM: 20 bytes (standard controller)
8 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Sequencing Mode

WP Tech Stencil:
Timer and Sequence Control

Properties

Table–5.276 Sequence (3) Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.

Table–5.276 Sequence (3) Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
SeqMod	Sequencing Mode	Class: Analog - Defines the sequencing mode, which determines how Outputs[1] to [3] operate as follows: • Linear (SeqMod = 0) causes the outputs to operate in a binary, linear sequence mode of operation. • Analog (SeqMod = 1) causes the outputs to operate in an analog sequence mode of operation. • Vernier (SeqMod = 2) causes Output[1] to operate in an analog mode, and the other outputs to supplement in a binary, linear sequence mode of operation.	0	0 - Linear 1 - Analog 2 - Vernier	A not active (NA) or a value outside the defined range results in the default selection of Linear.

Table–5.277 Sequence (3) Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
SeqEnb	Sequence Enable	Class: Digital - An ON or not active (NA) enables the Sequence (3) object algorithm. An OFF causes the outputs to be reset, where binary outputs are OFF and analog outputs are set to 0.0%, and the value of Stages On is set to 0 (zero).	OFF, ON, NA	
Input	Input	Class: Analog - The monitored value (0.0 to 100.0%) that determines the output action, which can range from full OFF to full ON conditions. A not active (NA) is evaluated the same as 0.0%, where all object outputs are set to OFF or 0.0%.	0.0 to 100.0%	
NumStgs	Number of Stages	Class: Analog - Defines the number of output stages used by the Sequence (3) object. The object algorithm scales the number of stages assigned to sequence ON over the full input range (0.0 to 100.0%). Outputs for stages not activated by the Number of Stages value (higher than) are set to not active (NA).	1, 2, or 3 If configured for Vernier, 2 is maximum	Typically set to 3. A number greater than 3 is evaluated as 3.

Table–5.278 Sequence (3) Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital / Analog - The first or lowest output stage if the Sequencing Mode is set to Linear or Analog, or the variable vernier output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear mode, or analog (0.0 to 100.0%) if Analog or Vernier.	Linear: OFF (0.0) ON (100.0) Analog or Vernier: 0.0 to 100.0%
Output[2]	Output [2]	Class: Digital / Analog - The second output stage if the Sequencing Mode is set to Linear or Analog, or the first binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[3]	Output [3]	Class: Digital / Analog - The third or highest output stage if the Sequencing Mode is set to Linear or Analog, or the second binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
StgsOn	Stages On	Class: Analog - If the Sequencing Mode is set to Linear or Analog, reflects the number of output stages either ON or above 0.0%. If the Sequencing Mode is set to Vernier, reflects the number of binary output stages ON only (2 maximum, not the Vernier portion).	Linear or Analog: 0, 1, 2, or 3 Vernier: 0, 1, or 2

Applying the Object

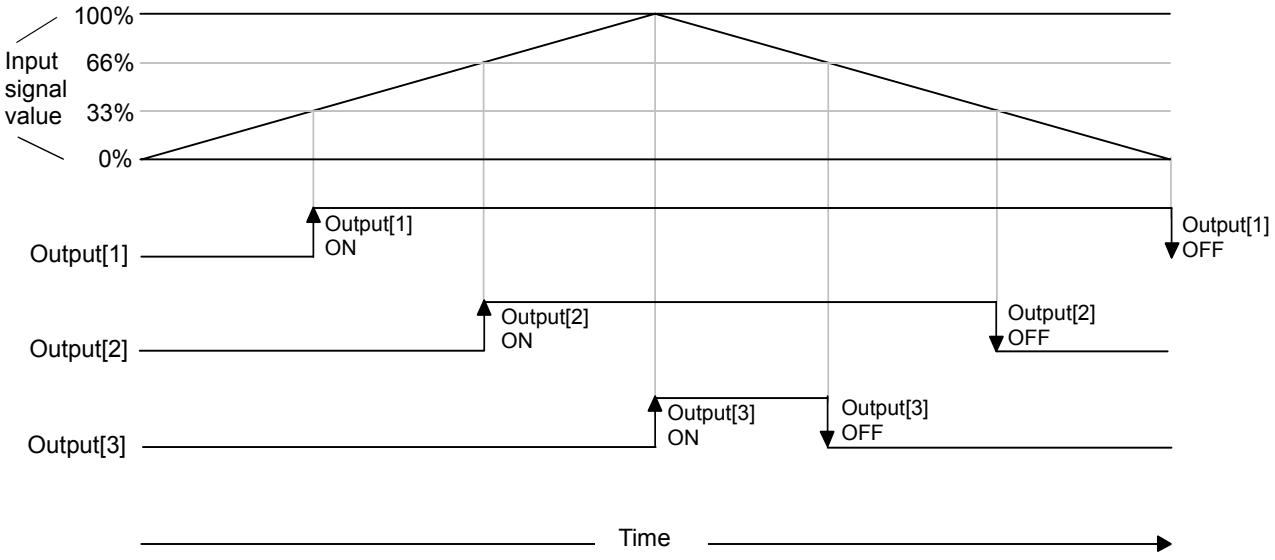
The setting of the Sequence Mode configuration property determines how Outputs[1], [2], and [3] operate in response to value changes at the Input (over the Input range of 0.0 to 100.0%). The three different modes are:

- Linear Sequence Mode.
- Analog Sequence Mode.
- Vernier Sequence Mode.

Linear Sequence Mode

The Linear sequence mode provides binary (OFF or ON) output stages that operate in a linear, first on / last off manner. The object algorithm divides the Number of Stages assigned (typically 3) equally between 0.0 and 100.0%. This establishes the ON values and OFF values for each stage that are compared to the Input value. The OFF value of a given stage is equal to the ON value of the previous stage, providing a fixed hysteresis between each stage.

Figure–5.149 shows the input to output function of a Sequence (3) object with a Linear sequence configuration and with the Number of Stages = 3.



Figure–5.149 Linear Configured Sequence (3) Object Input to Output Action (Number of Stages = 3).

As shown above, when the Input value moves upward from 0%:

- Output[1] stages ON when the Input value exceeds 33%.
- Output[2] stages ON when the Input value exceeds 66%.
- Output[3] stages ON when the Input value reaches 100%.

As the Input value moves downward from 100%:

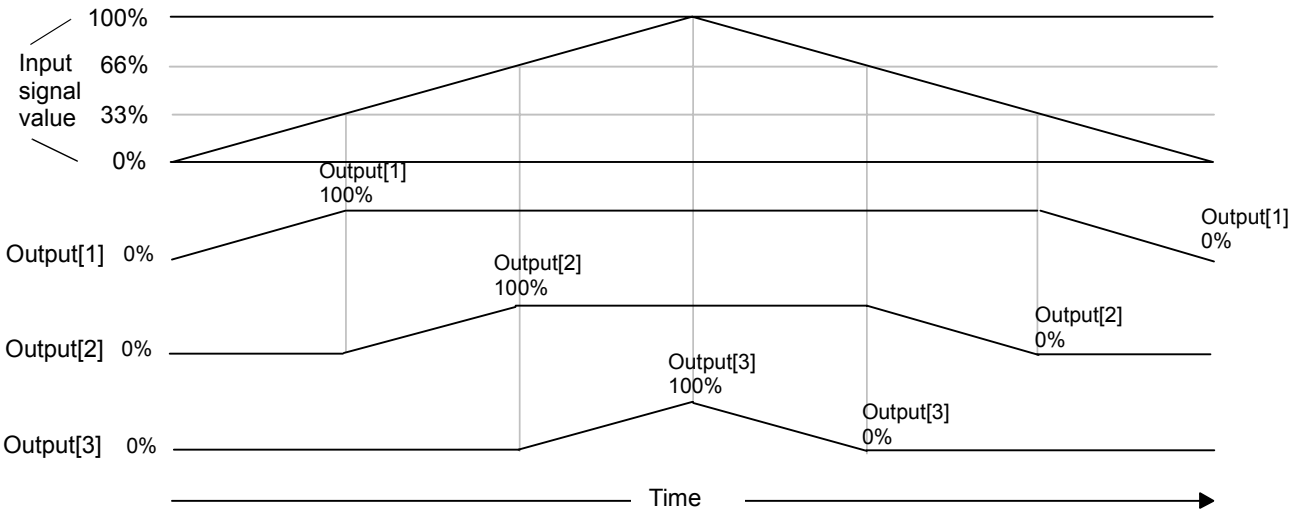
- Output[3] stages OFF when the Input value drops below 66%.
- Output[2] stages OFF when the Input value drops below 33%.
- Output[1] stages OFF when the Input value reaches 0%.

The Stages On output indicates the total number of outputs (0, 1, 2, or 3) that are currently ON.

Analog Sequence Mode

The Analog sequence mode provides analog (0.0 to 100.0%) output stages that operate in a proportionally sequenced manner. The object algorithm divides the Number of Stages assigned (typically 3) equally between 0.0 and 100.0%. This establishes the amount of the input range required to operate the individual 0 to 100% output range for each analog output. The Full Off value (0%) of a given stage is equal to the Full On (100%) of the previous stage, which provides a continuous analog sequencing effect between stages.

Figure–5.150 shows the input to output function of a Sequence (3) object with an Analog sequence configuration and with the Number of Stages = 3.



Figure–5.150 Analog Configured Sequence (3) Object Input to Output Action (Number of Stages = 3).

As shown above, when the Input value moves upward from 0%:

- Output[1] ranges between 0% and 100% as the Input value ranges between 0 and 33%, remaining at 100% above a 33% Input value.
- Output[2] ranges between 0% and 100% as the Input value ranges between 33% and 66%, remaining at 100% above a 66% Input value.
- Output[3] ranges between 0% and 100% as the Input value ranges between 66% and 100%.

As the Input value moves downward from 100%:

- Output[3] ranges between 100% and 0% as the Input value ranges between 100% and 66%, remaining at 0% below a 66% Input value.
- Output[2] ranges between 100% and 0% as the Input value ranges between 66% and 33%, remaining at 0% below a 33% Input value.
- Output[1] ranges between 100% and 0% as the Input value ranges between 33% and 0%.

The Stages On output indicates the total number of outputs (0, 1, 2, or 3) that currently have an output greater than 0%.

Vernier Sequence Mode

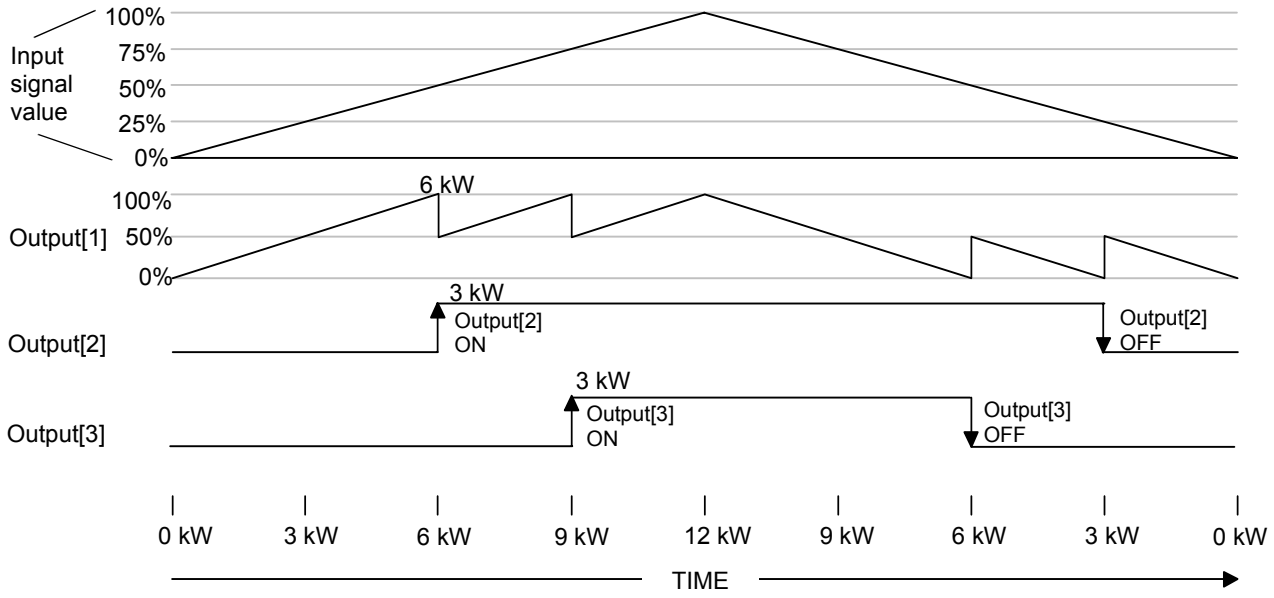
The Vernier sequence mode provides complete 0 to 100% proportional output control by using a combination of a single analog (0.0 to 100.0%) output stage along with additional binary (OFF or ON) output stages. Outputs operate in a proportionally sequenced manner.

The object algorithm divides the Number of Stages assigned plus two (the equivalent vernier output weight) equally between 0.0 and 100.0%. This establishes the ON values and OFF values for each binary stage, as compared to the Input value. The vernier (analog) Output[1] acts as the primary output, providing the proportional component in sequence with the control of the binary outputs (Outputs[2] and [3]).

Figure–5.151 shows the input to output function of a Sequence (3) object with a Vernier sequence configuration and with the Number of Stages = 2 (the number of digital stages used). In this example, the outputs control electric heat loads sized as follows:

- Output[1] (vernier) - 6 kW load with SCR (proportional) control.
- Output[2] (binary) - 3 kW load digitally controlled (OFF or ON).
- Output[3] (binary) - 3 kW load digitally controlled (OFF or ON).

This combination of loads with the vernier-configured Sequence (3) object provides a proportional output control sequence between 0 and 12 kW.



Figure–5.151 Vernier Configured Sequence (3) Object Input to Output Action (Number of Stages = 2).

As shown above, when the Input value moves upward from 0%:

- Output[1] ranges between 0% and 100% as the Input value ranges between 0 and 50%.
- Output[2] stages ON when the Input value reaches 50%, and Output[1] resets to 50%. As the Input value ranges between 50% and 75%, Output[1] ranges between 50% and 100%.
- Output[3] stages ON when the Input value reaches 75%, and Output[1] resets to 50%. As the Input value ranges between 75% and 100%, Output[1] ranges between 50% and 100%.

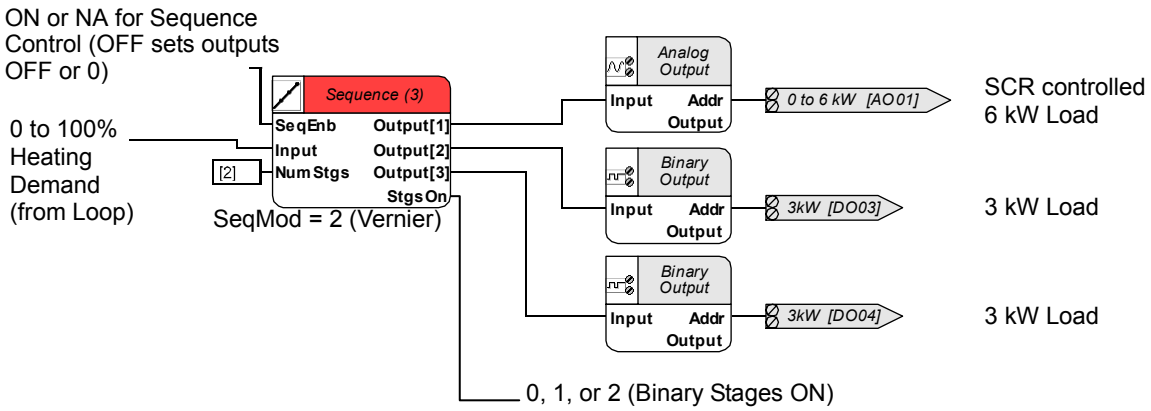
As the Input value moves downward from 100%:

- Output[1] ranges between 100% and 0% as the Input value ranges between 100% and 50%.
- Output[3] stages OFF when the Input value ranges reaches 50%, and Output[1] resets to 50%. As the Input value ranges between 50% and 25%, Output[1] ranges between 50% and 0%.
- Output[2] stages OFF when the Input value ranges reaches 25%, and Output[1] resets to 50%. As the Input value ranges between 25% and 0%, Output[1] ranges between 50% and 0%.

The Stages On output shows the total number of binary outputs that are currently ON (0, 1, or 2 only), as the vernier (analog) output (Output[1]) is not included.

Example

The Sequence (3) object shown in the example below Figure-5.152 performs the vernier output sequencing for the electric heating application previously described in Figure–5.151.



Figure–5.152 Example Sequence (3) Object in a Vernier (Electric Heating) Application.

Sequence (6)

Object Usage: The Sequence (6) object provides a method to sequence outputs based on the proportional 0.0 to 100.0% signal received at the object Input. Outputs[1] to [6] can function as either binary outputs (OFF or ON) or as analog outputs (0.0 to 100.0%), or as a combination of the two, depending on the Sequence Mode setting. The Sequence Mode can be set to one of the following:

- Linear Sequence Mode - outputs operate in a binary (OFF or ON), first-on / last-off logical sequence.
- Analog Sequence Mode - outputs operate in an analog (0.0 to 100.0%) continuously proportional sequence.
- Vernier Sequence Mode - a single output operates in a proportionally analog fashion, while the remaining five outputs offer additional binary (OFF or ON) sequence control.

The Sequence (6) object operates like the Sequence (3) (page 474) and Sequence (10) (page 488) objects, but with 6 total outputs. The Sequence (6) object may be paired with a Binary Encoder object (page 161) to provide 6 discrete levels of output using only three specially sized, digitally controlled loads.

Device Support: (all firmware revisions)

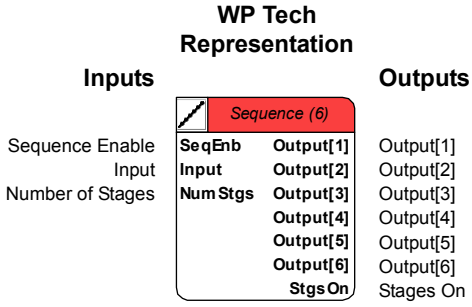
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 12 bytes
RAM: 26 bytes (standard controller)
14 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Sequencing Mode

WP Tech Stencil:
Timer and Sequence Control

Properties

Table–5.279 Sequence (6) Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
SeqMod	Sequencing Mode	Class: Analog - Defines the sequencing mode, which determines how Outputs[1] to [6] operate, as follows: • Linear (SeqMod = 0) causes the outputs to operate in a binary, linear sequence mode of operation. • Analog (SeqMod = 1) causes the outputs to operate in an analog sequence mode of operation. • Vernier (SeqMod = 2) causes Output[1] to operate in an analog mode, and the other outputs to supplement in a binary, linear sequence mode of operation.	0	0 - Linear 1 - Analog 2 - Vernier	A not active (NA) or a value outside the defined range results in the default selection of Linear.

Table–5.280 Sequence (6) Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
SeqEnb	Sequence Enable	Class: Digital - An ON or not active (NA) enables the Sequence (6) object algorithm. An OFF causes the outputs to be reset, where binary outputs are OFF and analog outputs are set to 0.0%, and the value of Stages On is set to 0 (zero).	OFF, ON, NA	
Input	Input	Class: Analog - The monitored value (0.0 to 100.0%) that determines the output action, which can range from full OFF to full ON conditions. A not active (NA) is evaluated the same as 0.0%, where all object outputs are set to OFF or 0.0%.	0.0 to 100.0%	
NumStgs	Number of Stages	Class: Analog - Defines the number of output stages used by the Sequence (6) object. The object algorithm scales the number of stages assigned to sequence ON over the full input range (0.0 to 100.0%). Outputs for stages not activated by the Number of Stages value (higher than) are set to not active (NA).	1, 2, 3, 4, 5, or 6 If configured for Vernier, 5 is maximum	A number greater than 6 is evaluated as 6.

Table–5.281 Sequence (6) Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital / Analog - The first or lowest output stage if the Sequencing Mode is set to Linear or Analog, or the variable vernier output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear mode, or analog (0.0 to 100.0%) if Analog or Vernier.	Linear: OFF (0.0) ON (100.0) Analog or Vernier: 0.0 to 100.0%
Output[2]	Output [2]	Class: Digital / Analog - The second output stage if the Sequencing Mode is set to Linear or Analog, or the first binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[3]	Output [3]	Class: Digital / Analog - The third output stage if the Sequencing Mode is set to Linear or Analog, or the second binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[4]	Output [4]	Class: Digital / Analog - The fourth output stage if the Sequencing Mode is set to Linear or Analog, or the third binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[5]	Output [5]	Class: Digital / Analog - The fifth output stage if the Sequencing Mode is set to Linear or Analog, or the fourth binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[6]	Output [6]	Class: Digital / Analog - The sixth or highest output stage if the Sequencing Mode is set to Linear or Analog, or the fifth binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
StgsOn	Stages On	Class: Analog - If the Sequencing Mode is set to Linear or Analog, reflects the number of output stages either ON or above 0.0%. If the Sequencing Mode is set to Vernier, reflects the number of binary output stages ON only (5 maximum, not the Vernier portion).	Linear or Analog: 0, 1, 2, 3, 4, 5, or 6 Vernier: 0, 1, 2, 3, 4, or 5

Applying the Object

The setting of the Sequence Mode configuration property determines how Outputs[1] through [6] operate in response to value changes at the Input (over the Input range of 0.0 to 100.0%). The three different modes are:

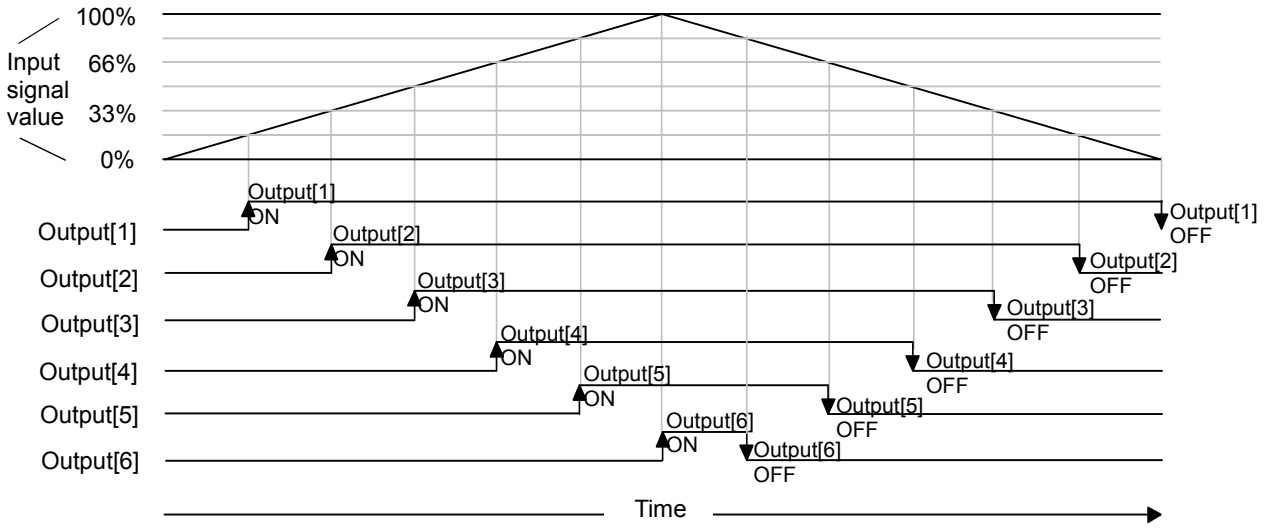
- Linear Sequence Mode.
- Analog Sequence Mode.
- Vernier Sequence Mode.

Linear Sequence Mode

The Linear sequence mode provides binary (OFF or ON) output stages that operate in a linear, first on / last off manner. The object algorithm divides the Number of Stages assigned (maximum of 6) equally between 0.0 and 100.0%. This establishes the ON values and OFF values for each stage that are compared to the Input value.

The OFF value of a given stage is equal to the ON value of the previous stage, providing a fixed hysteresis between each stage.

Figure–5.153 shows the input to output function of a Sequence (6) object with a Linear sequence configuration and with the Number of Stages = 6.



Figure–5.153 Linear Configured Sequence (6) Object Input to Output Action (Number of Stages = 6).

As shown above, when the Input value moves upward from 0%, Outputs[1] through [6] stage ON based on the calculated ON values from the following equation:

Stage ON value = [(100% ÷ Number of Stages) x actual stage number]

For the example shown above, where Number of Stages = 6, the first stage (Output[1]) stages ON when the Input value is greater than 16.6%.

Stage 1 ON value = [(100% ÷ 6) x 1] = 16.6%

As the Input continues to increase, additional outputs stage ON in equal increments (33%, 50%, etc.), with the highest valid output (typically 6) staged ON at a full Input condition of 100%.

As the Input value moves downward from 100%, the ON value of the previous stage becomes the OFF value for the current stage. For the example shown above, Output[6] stages OFF when the Input value drops below 83.3%. This pattern continues until the Input reaches 0%, where Output[1] stages OFF.

The Stages On output indicates the total number of outputs (0, 1, 2, 3, 4, 5, or 6) that are currently ON.

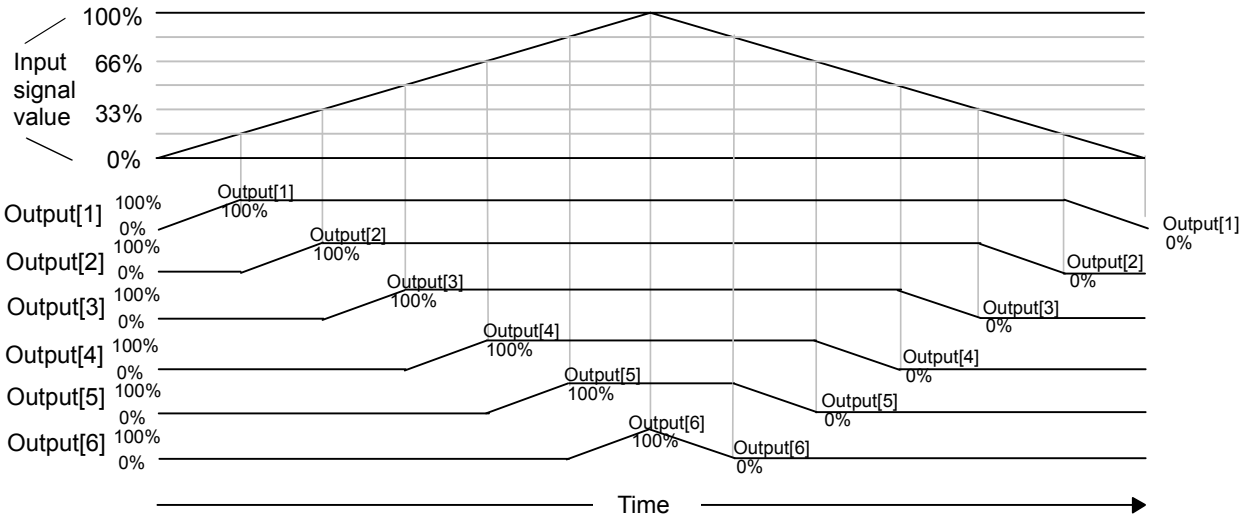
Use with Binary Encoder Object

A Linear configured Sequence (6) object can be paired with a Binary Encoder object to provide 6 discrete levels of output using only three specially-sized digitally controlled loads. Refer to the Binary Encoder object for more details on this type of application.

Analog Sequence Mode

The Analog sequence mode provides analog (0.0 to 100.0%) output stages that operate in a proportionally sequenced manner. The object algorithm divides the Number of Stages assigned (maximum of 6) equally between 0.0 and 100.0%. This establishes the amount of the input range required to operate the individual 0 to 100% output range for each analog output. The Full Off value (0%) of a given stage is equal to the Full On (100%) of the previous stage, which provides a continuous analog sequencing effect between stages.

Figure–5.154 shows the input to output function of a Sequence (6) object with an Analog sequence configuration and with the Number of Stages = 6.



Figure–5.154 Analog Configured Sequence (3) Object Input to Output Action (Number of Stages = 6).

As shown above, when the Input value moves upward from 0%, Outputs[1] through [6] range from 0% to 100% based on the calculated Full On (100%) point for each stage, derived from the following equation:

Full Stage output = [(100% ÷ Number of Stages) x actual stage number]

For the example shown above, where Number of Stages = 6, the first stage (Output[1]) is at 100% when the Input value reaches 16.6%.

Full Stage 1 output = [(100% ÷ 6) x 1] = 16.6%

As the Input continues to increase, additional outputs range from 0% to 100% in proportion to the Input signal, with the highest valid output (typically 6) at 100% (Full On) at the full Input condition of 100%.

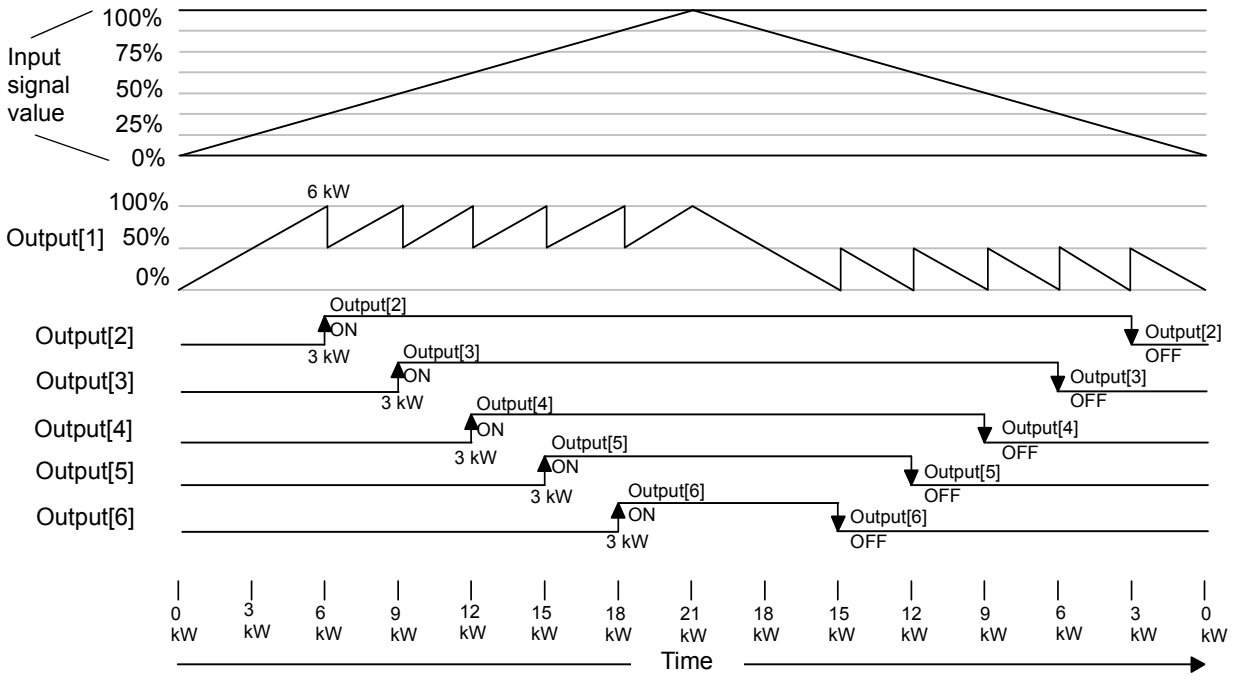
As the Input value moves downward from 100%, the outputs mirror their previous 100% to 0% output range as before, until Output[1] ranges to 0% when the Input value reaches 0.0%.

The Stages On output indicates the total number of outputs (0, 1, 2, 3, 4, 5, or 6) that currently have an output greater than 0%.

Vernier Sequence Mode

The Vernier sequence mode provides complete 0 to 100% proportional output control by using a combination of a single analog (0.0 to 100.0%) output stage along with additional binary (OFF or ON) output stages. Outputs operate in a proportionally sequenced manner.

The object algorithm divides the Number of Stages assigned plus two (the equivalent vernier output weight) equally between 0.0 and 100.0%. This establishes the ON values and OFF values for each binary stage, as compared to the Input value. The vernier (analog) Output[1] acts as the primary output, providing the proportional component in sequence with the control of the binary outputs (Outputs[2] through [6]).



Figure–5.155 Vernier Configured Sequence (6) Object Input to Output Action (Number of Stages = 5).

Figure–5.155 shows the input to output function of a Sequence (6) object with a Vernier sequence configuration and with the Number of Stages = 5 (the number of digital stages used). In this example, the outputs control electric heat loads sized as follows:

- Output[1] (vernier) - a 6 kW load with SCR (proportional) control.
- Outputs[2] through [6] (binary) - each a 3 kW load digitally controlled (OFF or ON).

This combination of loads with the vernier-configured Sequence (6) object provides a proportional output control sequence between 0 and 21 kW.

As shown above, when the Input value moves upward from 0%:

- Output[1] ranges between 0% and 100% as the Input value ranges between 0 and 28.6% (2 / 7).
- Output[2] stages ON when the Input value reaches 28.6%, and Output[1] resets to 50%. Further increases in the Input value modulate Output[1] through the 50% to 100% range and stage ON the remaining Outputs[3] through [6] as shown.

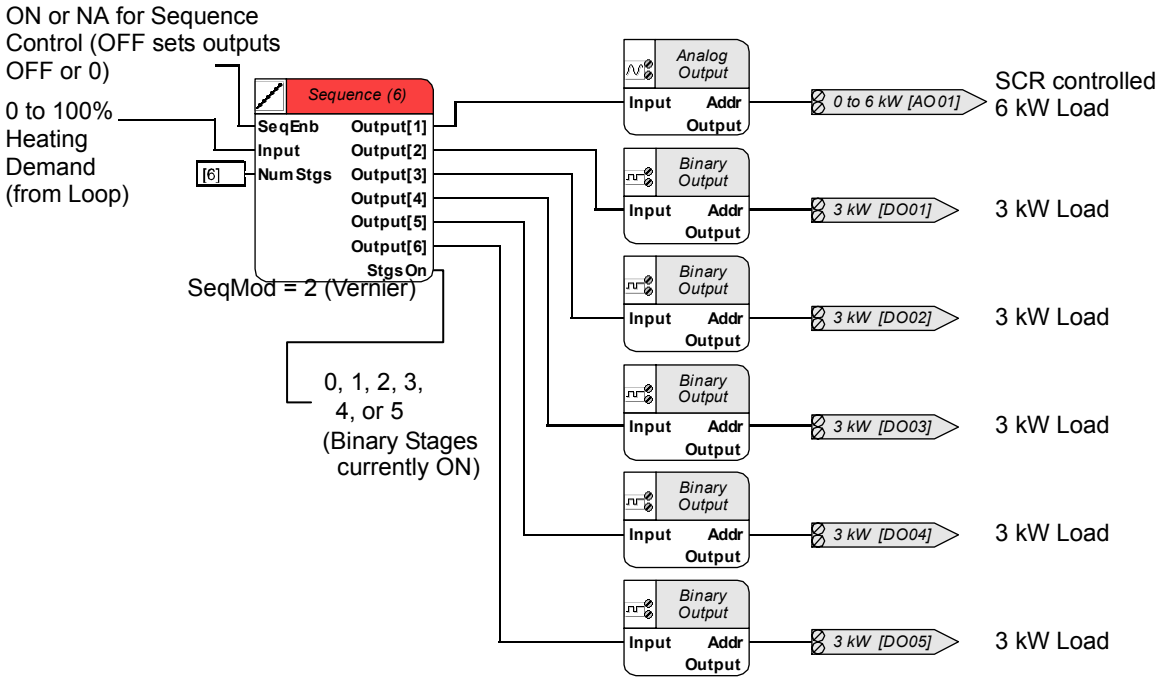
As the Input value decreases from 100%:

- Output[1] ranges between 100% and 0% as the Input value ranges between 100% and 71.4% (5 / 7).
- Output[6] stages OFF when the Input value ranges reaches 71.4%, and Output[1] resets to 50%. Further decreases in the Input value modulate Output[1] through the 50% to 0% range and stage OFF the remaining Outputs[5] through [2] as shown.

The Stages On output shows the total number of binary outputs that are currently ON (0, 1, 2, 3, 4, or 5 only), as the vernier (analog) output (Output[1]) is not included.

Example

The Sequence (6) object shown in Figure–5.156 performs the vernier output sequencing for the electric heating application described in Figure–5.155.



Figure–5.156 Example Sequence (6) Object in a Vernier (Electric Heating) Application.

Sequence (10)

Object Usage: The Sequence (10) object provides a method to sequence up to 10 outputs based on the proportional 0.0 to 100.0% signal received at the object Input. Outputs[1] to [10] can function as either binary outputs (OFF or ON) or as analog outputs (0.0 to 100.0%), or as a combination of the two, depending on the Sequence Mode setting. The Sequence Mode can be set to one of the following:

- Linear Sequence Mode - outputs operate in a binary (OFF or ON), first-on / last-off logical sequence.
- Analog Sequence Mode - outputs operate in an analog (0.0 to 100.0%) continuously proportional sequence.
- Vernier Sequence Mode - a single output operates in a proportionally analog fashion, while the remaining nine outputs offer additional binary (OFF or ON) sequence control.

The Sequence (10) object operates like the Sequence (3) (page 474) and Sequence (6) (page 481) objects, but has the most total outputs.

Device Support: (all firmware revisions)

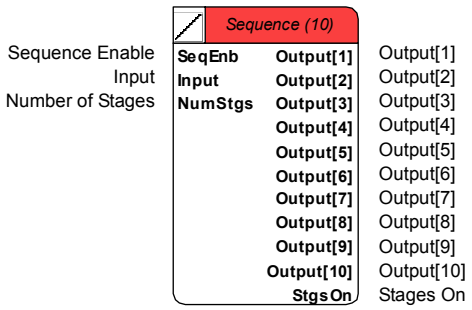
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 12 bytes
RAM: 34 bytes (standard controller)
22 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time
Sequencing Mode

WP Tech Stencil:
Timer and Sequence Control

Properties

Table–5.282 Sequence (10) Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.

Table–5.282 Sequence (10) Object Configuration Properties. (Continued)

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
SeqMod	Sequencing Mode	Class: Analog - Defines the sequencing mode, which determines how Outputs[1] to [10] operate, as follows: • Linear (SeqMod = 0) causes the outputs to operate in a binary, linear sequence mode of operation. • Analog (SeqMod = 1) causes the outputs to operate in an analog sequence mode of operation. • Vernier (SeqMod = 2) causes Output[1] to operate in an analog mode, and the other outputs to supplement in a binary, linear sequence mode of operation.	0	0 - Linear 1 - Analog 2 - Vernier	A not active (NA) or a value outside the defined range results in the default selection of Linear.

Table–5.283 Sequence (10) Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
SeqEnb	Sequence Enable	Class: Digital - An ON or not active (NA) enables the Sequence (10) object algorithm. An OFF causes the outputs to be reset, where binary outputs are OFF and analog outputs are set to 0.0%, and the value of Stages On is set to 0 (zero).		
Input	Input	Class: Analog - The monitored value (0.0 to 100.0%) that determines the output action, which can range from full OFF to full ON conditions. A not active (NA) is evaluated the same as 0.0%, where all object outputs are set to OFF or 0.0%.	0.0 to 100.0%	
NumStgs	Number of Stages	Class: Analog - Defines the number of output stages used by the Sequence (10) object. The object algorithm scales the number of stages assigned to sequence ON over the full input range (0.0 to 100.0%). Outputs for stages not activated by the Number of Stages value (higher than) are set to not active (NA).	1 through 10, (integer) If configured for Vernier, 9 is maximum	A number greater than 10 is evaluated as 10.

Table–5.284 Sequence (10) Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital / Analog - The first or lowest output stage if the Sequencing Mode is set to Linear or Analog, or the variable vernier output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear mode, or analog (0.0 to 100.0%) if Analog or Vernier.	Linear: OFF (0.0) ON (100.0) Analog or Vernier: 0.0 to 100.0%

Table–5.284 Sequence (10) Object Output Properties. (Continued)

Abbrev.	Name	Class / Description	Valid Values
Output[2]	Output [2]	Class: Digital / Analog - The second output stage if the Sequencing Mode is set to Linear or Analog, or the first binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[3]	Output [3]	Class: Digital / Analog - The third output stage if the Sequencing Mode is set to Linear or Analog, or the second binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[4]	Output [4]	Class: Digital / Analog - The fourth output stage if the Sequencing Mode is set to Linear or Analog, or the third binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[5]	Output [5]	Class: Digital / Analog - The fifth output stage if the Sequencing Mode is set to Linear or Analog, or the fourth binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[6]	Output [6]	Class: Digital / Analog - The sixth output stage if the Sequencing Mode is set to Linear or Analog, or the fifth binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[7]	Output [7]	Class: Digital / Analog - The seventh output stage if the Sequencing Mode is set to Linear or Analog, or the sixth binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[8]	Output [8]	Class: Digital / Analog - The eighth output stage if the Sequencing Mode is set to Linear or Analog, or the seventh binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[9]	Output [9]	Class: Digital / Analog - The ninth output stage if the Sequencing Mode is set to Linear or Analog, or the eighth binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
Output[10]	Output [10]	Class: Digital / Analog - The tenth or highest output stage if the Sequencing Mode is set to Linear or Analog, or the ninth binary output if the Sequencing Mode is set to Vernier. This output is binary (OFF/ON) if Linear or Vernier mode, or analog (0.0 to 100.0%) if Analog mode.	Linear or Vernier: OFF (0.0) ON (100.0) Analog: 0.0 to 100.0%
StgsOn	Stages On	Class: Analog - If the Sequencing Mode is set to Linear or Analog, reflects the number of output stages either ON or above 0.0%. If the Sequencing Mode is set to Vernier, reflects the number of binary output stages ON only (9 maximum, not the Vernier portion).	Linear or Analog: 0 to 10, integer Vernier: 0 to 9, integer

Applying the Object

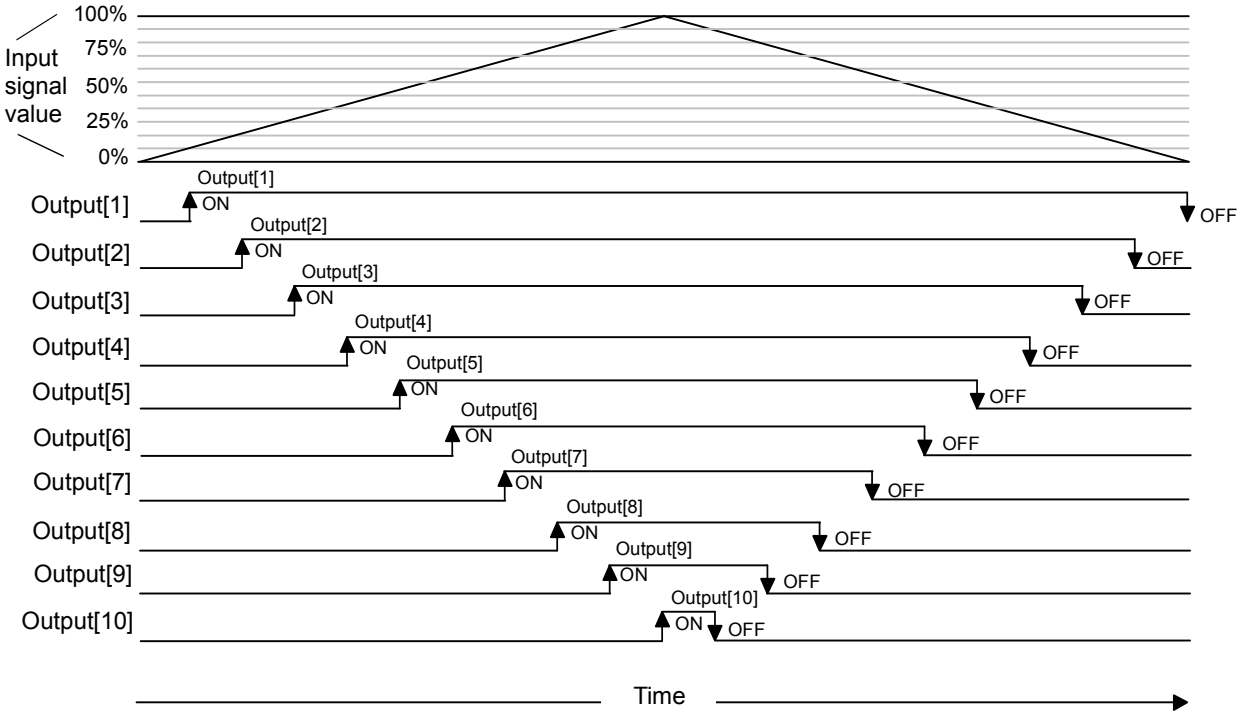
The setting of the Sequence Mode configuration property determines how Outputs[1] through [10] operate in response to value changes at the Input (over the Input range of 0.0 to 100.0%). The three different modes are:

- Linear Sequence Mode.
- Analog Sequence Mode.
- Vernier Sequence Mode.

Linear Sequence Mode

The Linear sequence mode provides binary (OFF or ON) output stages that operate in a linear, first on / last off manner. The object algorithm divides the Number of Stages assigned (maximum of 10) equally between 0.0 and 100.0%. This establishes the ON values and OFF values for each stage that are compared to the Input value. The OFF value of a given stage is equal to the ON value of the previous stage, providing a fixed hysteresis between each stage.

Figure–5.157 shows the input to output function of a Sequence (10) object with a Linear sequence configuration and with the Number of Stages = 10.



Figure–5.157 Linear Configured Sequence (10) Object Input to Output Action (Number of Stages = 10).

As shown, when the Input value moves upward from 0%, Outputs[1] through [10] stage ON based on the calculated ON values from the following equation:

Stage ON value = [(100% ÷ Number of Stages) x actual stage number]

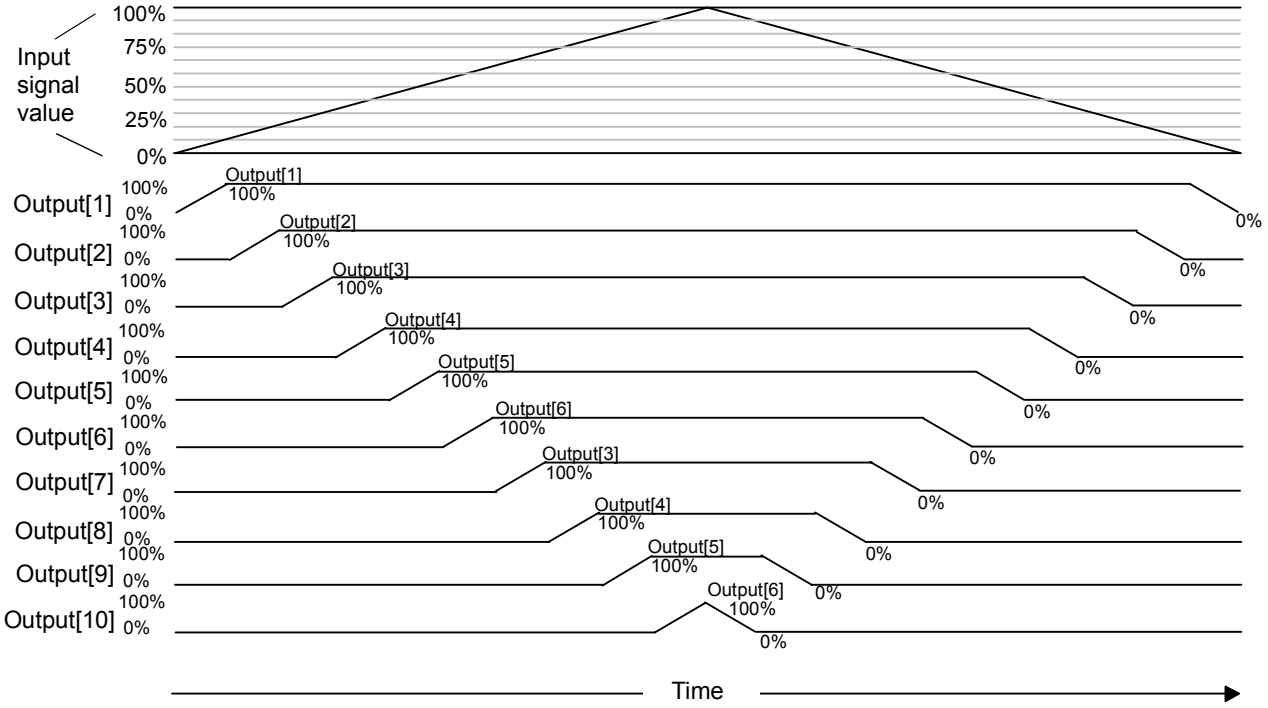
In this example, where Number of Stages = 10, the first stage (Output[1]) stages ON when the Input value is greater than 10.0%. As the Input continues to increase, additional outputs stage ON in equal increments (20%, 30%, etc.), with the highest valid output (maximum 10) staged ON at a full Input condition of 100%.

As the Input value moves downward from 100%, the ON value of the previous stage becomes the OFF value for the current stage. For the example shown above, Output[10] stages OFF when the Input value drops below 90.0%. This pattern continues until the Input reaches 0%, where Output[1] stages OFF. The Stages On output indicates the total number of outputs (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, or 10) that are currently ON.

Analog Sequence Mode

The Analog sequence mode provides analog (0.0 to 100.0%) output stages that operate in a proportionally sequenced manner. The object algorithm divides the Number of Stages assigned (maximum of 10) equally between 0.0 and 100.0%. This establishes the amount of the input range required to operate the individual 0 to 100% output range for each analog output. The Full Off value (0%) of a given stage is equal to the Full On (100%) of the previous stage, which provides a continuous analog sequencing effect between stages.

Figure–5.158 shows the input to output function of a Sequence (10) object with an Analog sequence configuration and with the Number of Stages = 10.



Figure–5.158 Analog Configured Sequence (10) Object Input to Output Action (Number of Stages = 10).

As shown, when the Input value moves upward from 0%, Outputs[1] through [10] range from 0% to 100% based on the calculated Full On (100%) point for each stage, derived from the following equation:

Full Stage Output = [(100% ÷ Number of Stages) x actual stage number]

For the example just shown, where Number of Stages = 10, the first stage (Output[1]) is at 100% when the Input value reaches 10.0%. As the Input continues to increase, additional outputs range from 0% to 100% in proportion to the Input signal, with the highest valid output (maximum 10) at 100% (Full On) at the full Input condition of 100%.

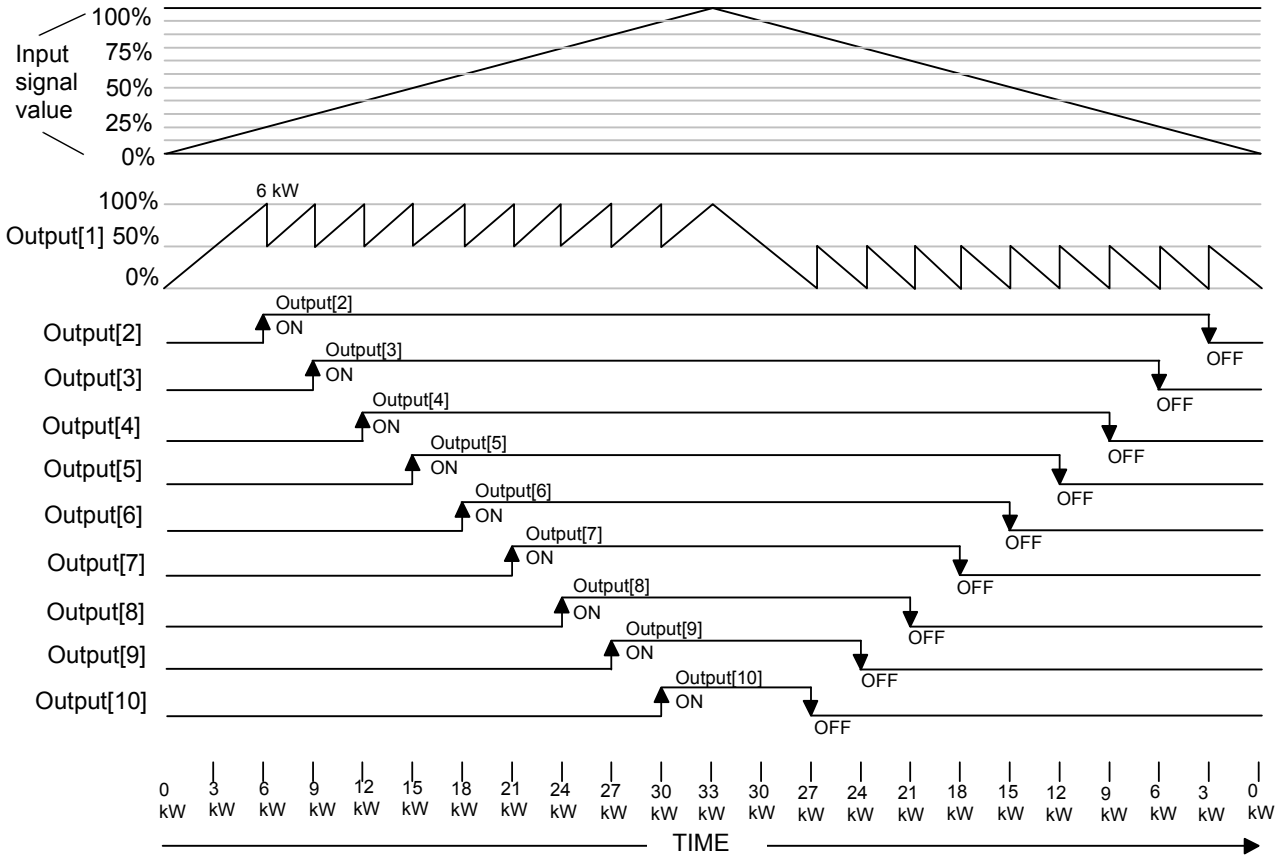
As the Input value moves downward from 100%, the outputs mirror their previous 100% to 0% output range as before, until Output[1] ranges to 0% when the Input value reaches 0.0%.

The Stages On output indicates the total number of outputs (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, or 10) that currently have an output greater than 0%.

Vernier Sequence Mode

The Vernier sequence mode provides complete 0 to 100% proportional output control by using a combination of a single analog (0.0 to 100.0%) output stage along with additional binary (OFF or ON) output stages. Outputs operate in a proportionally sequenced manner.

The object algorithm divides the Number of Stages assigned plus two (the equivalent vernier output weight) equally between 0.0 and 100.0%. This establishes the ON values and OFF values for each binary stage, as compared to the Input value. The vernier (analog) Output[1] acts as the primary output, providing the proportional component in sequence with the control of the binary outputs (Outputs[2] through [10]).



Figure–5.159 Vernier Configured Sequence (10) Object Input to Output Action (Number of Stages = 9).

The following diagram Figure-5.151 shows the input to output function of a Sequence (10) object with a Vernier sequence configuration and with the Number of Stages = 9 (the number of digital stages used). In this example, the outputs control electric heat loads sized as follows:

- Output[1] (vernier) - a 6 kW load with SCR (proportional) control.
- Outputs[2] through [10] (binary) - each a 3 kW load digitally controlled (OFF or ON).

This load combination with the vernier-configured Sequence (10) object provides a proportional output control sequence between 0 and 33 kW.

As shown in Figure–5.159, when the Input value moves upward from 0%:

- Output[1] ranges between 0% and 100% as the Input value ranges between 0 and 18.2% (2 / 11).
- Output[2] stages ON when the Input value reaches 18.2%, and Output[1] resets to 50%. Further increases in the Input value modulate Output[1] through the 50% to 100% range and stage ON the remaining Outputs[3] through [10] as shown.

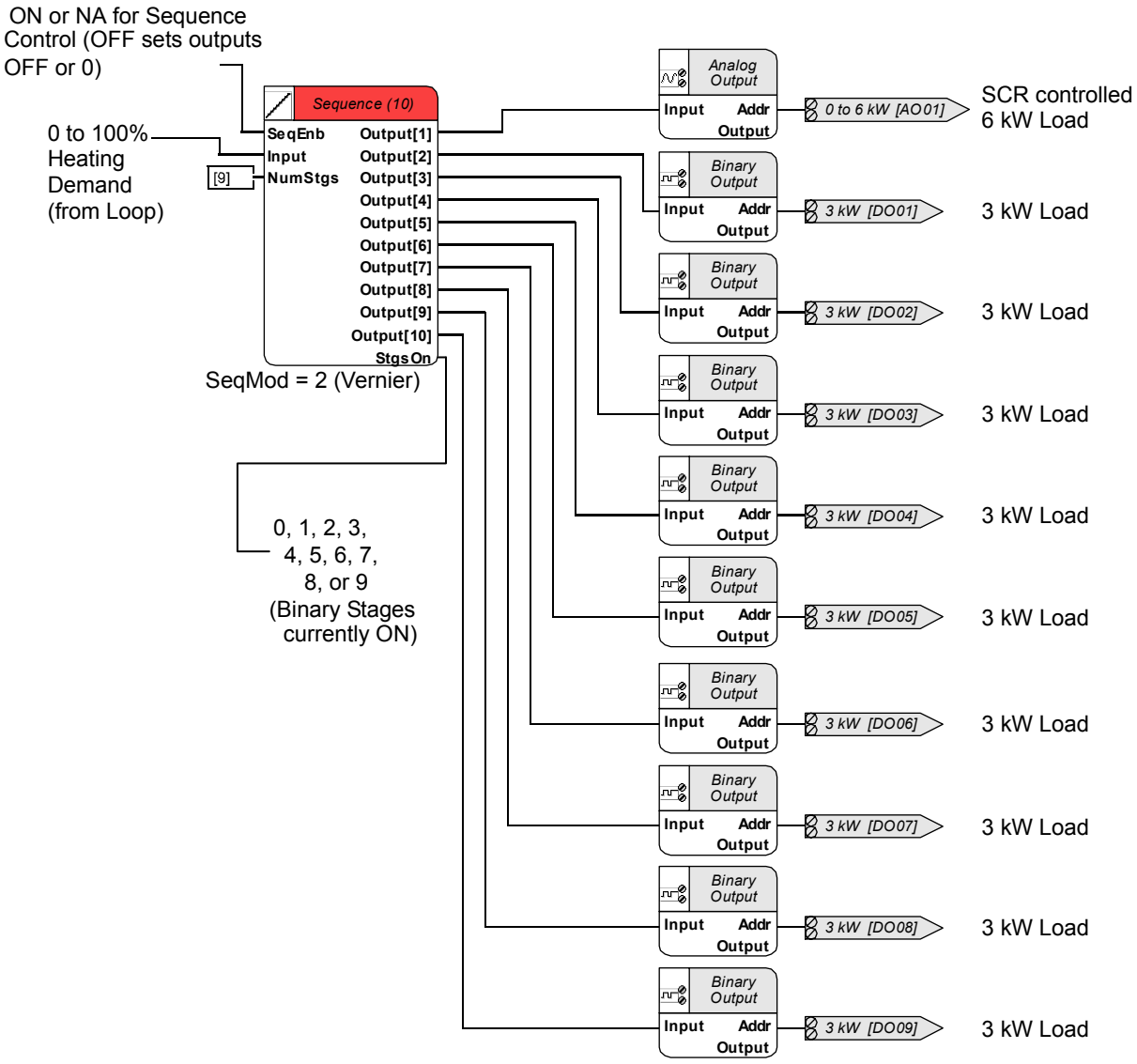
As the Input value decreases from 100%:

- Output[1] ranges between 100% and 0% as the Input value ranges between 100% and 81.8% (9 / 11).
- Output[10] stages OFF when the Input value ranges reaches 81.8%, and Output[1] resets to 50%. Further decreases in the Input value modulate Output[1] through the 50% to 0% range and stage OFF the remaining Outputs[9] through [2] as shown.

The Stages On output shows the total number of binary outputs that are currently ON (0, 1, 2, 3, 4, 5, 6, 7, 8, or 9 only), as the vernier (analog) output (Output[1]) is not included.

Example

The Sequence (10) object shown in Figure–5.160 below performs the vernier output sequencing for the electric heating application previously described in Figure–5.159.



Figure–5.160 Example Sequence (10) Object in a Vernier (Electric Heating) Application.

Setpoint Control

Object Usage: The Setpoint Control object generates a set of active setpoint output values based upon single or dual occupied setpoints, unoccupied setpoints, and the actual occupancy condition. This object is typically used to generate the occupied and unoccupied heating, cooling, and economizer setpoints from a single or dual setpoint input source.

Device Support: (all firmware revisions)

MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

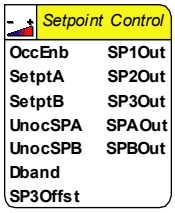
Memory Requirements: (per object)

EEPROM: 20 bytes
RAM: 30 bytes (standard controller)
10 bytes (MN 800)

WP Tech Representation

Inputs

Occupancy Enable
Setpoint A
Setpoint B
Unoccupied Setpoint A
Unoccupied Setpoint B
Deadband
Setpoint 3 Offset



Outputs

Setpoint Output 1
Setpoint Output 2
Setpoint Output 3
Setpoint Output A
Setpoint Output B

Configuration Properties

Object Name
Object Description
Process Time
Setpoint Type Select

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.285 Setpoint Control Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.
Type	Setpoint Type Select	Class: Analog - Determines whether the object algorithm uses either a single or a dual setpoint when calculating occupancy setpoints.	0	0 - Single 1 - Dual	

Table–5.286 Setpoint Control Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
OccEnb	Occupancy Enable	Class: Digital - An ON or not active (NA) establishes the occupied mode, whereby the object calculates and outputs occupied setpoints. An OFF establishes the unoccupied mode, whereby the object outputs unoccupied setpoints.	—	
SetptA	Setpoint A	Class: Analog - Defines either the occupied single setpoint, or the occupied cooling setpoint if configured for dual setpoints.	-163.83 to 16383	See the Single Setpoint mode and Dual Setpoint mode sections for details on how a not active (NA) condition for these properties affects the operation of the object.
SetptB	Setpoint B	Class: Analog - Defines the occupied heating setpoint (used only if configured for dual setpoints).		
UnocSPA	Unoccupied Setpoint A	Class: Analog - Defines the unoccupied cooling setpoint value.		
UnocSPB	Unoccupied Setpoint B	Class: Analog - Defines the unoccupied heating setpoint value.		
Dband	Deadband	Class: Analog - Defines the Setpoint Controller deadband, used differently according to the object's Setpoint Type Select configuration. • If a Single setpoint, Deadband is used to calculate occupied setpoint output values. • If Dual setpoints, Deadband is used to prevent occupied setpoint crossover.		
SP3Offst	Setpoint 3 Offset	Class: Analog - Defines an offset reference value subtracted from the cooling setpoint (SetptA) to generate the economizer setpoint (SP3Out).		

Table–5.287 Setpoint Control Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
SP1Out	Setpoint Output 1	Class: Analog - The active setpoint 1 output value typically defined as the cooling setpoint.	-163.83 to 16383
SP2Out	Setpoint Output 2	Class: Analog - The active setpoint 2 output value typically defined as the heating setpoint.	
SP3Out	Setpoint Output 3	Class: Analog - The active setpoint 3 output value typically defined as the calculated economizer setpoint.	
SPAOut	Setpoint Output A	Class: Analog - The calculated occupied cooling setpoint value. Typically used to provide occupancy setpoint targets for the optimum start / stop function.	-163.83 to 16383
SPBOut	Setpoint Output B	Class: Analog - The calculated occupied heating setpoint value. Typically used to provide occupancy setpoint targets for the optimum start / stop function.	

Applying the Object

The Setpoint Control object is typically used for generating temperature setpoints for applications with occupied / unoccupied control periods. Three separate outputs offer active setpoints typically used for cooling, heating, and economizer modes of operation. These outputs are often connected to matching inputs of a Loop Sequenced object. Two additional outputs of the Setpoint Control object represent the calculated occupied cooling and heating setpoints, typically used with an Optimized Start / Stop object.

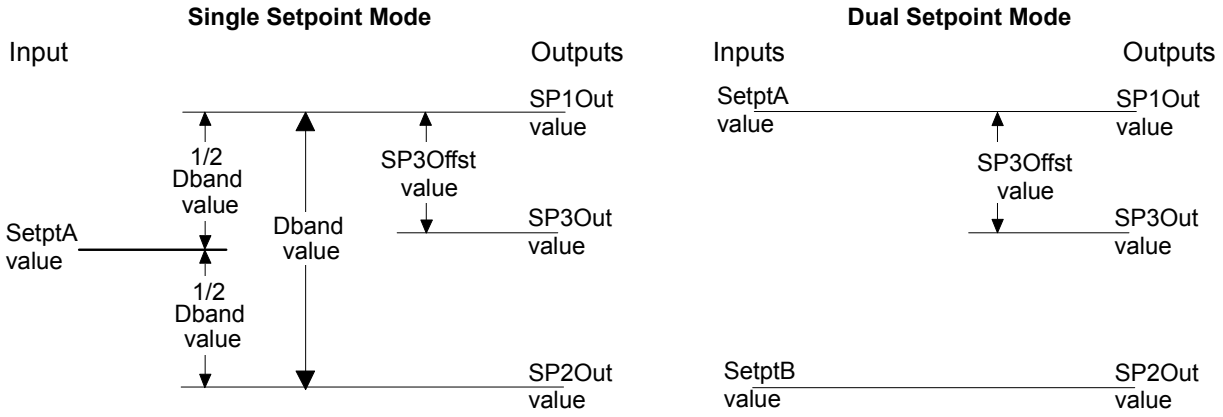
Single Setpoint Mode vs. Dual Setpoint Mode

The Setpoint Control object features two configurations modes, single setpoint mode or dual setpoint mode, determined by the configuration property Setpoint Type Select. The two modes function differently.

Selection of single setpoint mode causes the Setpoint Control object to calculate the active occupied setpoint outputs for SP1Out, SP2Out, and SP3Out from a single setpoint input SetptA value and the Dband value. The SetptB input is not evaluated. Unoccupied setpoint values are derived from UnocSPA and UnocSPB.

Selection of dual setpoint mode causes the Setpoint Control object to calculate the active occupied setpoint outputs for SP1Out, SP2Out, and SP3Out from separate inputs SetptA and SetptB. The Dband value is used to prevent occupied setpoint crossover. As in the single setpoint mode, unoccupied setpoint values are derived from UnocSPA and UnocSPB.

Figure–5.161 compares the two configuration modes during an occupied period. Both configuration modes are explained in the following sections.



Figure–5.161 Comparison of Single Setpoint and Dual Setpoint Modes in Occupied Period.

Single Setpoint Mode

Occupied Mode

When Occupancy Enable = ON or not active (NA), the Setpoint Control object calculates the output setpoint values as follows:

SP1Out = SetptA + (Dband ÷ 2)
SP2Out = SetptA - (Dband ÷ 2)
SP3Out = [SetptA + (Dband ÷ 2)] - SP3Offst

SPAOut = SetptA + (Dband ÷ 2)
SPBOut = SetptA - (Dband ÷ 2)

A not active (NA) value at these object inputs produces these results:

- If SetptA is NA, all object outputs are set to NA.
- If Dband is a negative value or NA, the value 0.0 is used as Dband.
- If SP3Offst is a negative value or NA, the value 0.0 is used as SP3Offst.

Unoccupied Mode

When Occupancy Enable = OFF, the Setpoint Control object calculates the output setpoint values as follows:

SP1Out = UnoccSPA
SP2Out = UnoccSPB
SP3Out = UnoccSPA - SP3Offst

SPAOut = SetptA + (Dband ÷ 2)
SPBOut = SetptA - (Dband ÷ 2)

Unoccupied setpoint crossover is prevented by comparing UnocSPB and UnocSPA values.

If UnocSPB > UnocSPA, the UnocSPB = UnocSPA.

A not active (NA) value at these object inputs produces these results:

- If SetptA is NA, SPAOut and SPBOut are set to NA.
- If UnoccSPA is NA, SP1Out and SP3Out are set to NA.
- If UnoccSPB is NA, SP2Out is set to NA.
- If Dband is a negative value or NA, the value 0.0 is used as Dband.
- If SP3Offst is a negative value or NA, the value 0.0 is used as SP3Offst.

Dual Setpoint Mode

Occupied Mode

When Occupancy Enable = ON or not active (NA), the Setpoint Control object calculates the output setpoint values as follows:

SP1Out = SetptA
SP2Out = SetptB
SP3Out = SetptA - SP3Offst

SPAOut = SetptA
SPBOut = SetptB

Occupied setpoint crossover is prevented by comparison of the values for SetptA, SetptB, and Dband.

If SetptB > (SetptA- Dband), then SetptB = (SetptA - Dband).

A not active (NA) value at these object inputs produces these results:

- If SetptA is NA, SP1Out, SP3Out, and SPAOut are set to NA.
- If SetptB is NA, SP2Out and SPBOut are set to NA.
- If Dband is a negative value or NA, the value 0.0 is used as Dband.
- If SP3Offst is a negative value or NA, the value 0.0 is used as SP3Offst.

Unoccupied Mode

When Occupancy Enable = OFF, the Setpoint Control object calculates the output setpoint values as follows:

SP1Out = UnoccSPA
SP2Out = UnoccSPB
SP3Out = UnoccSPA - SP3Offst

SPAOut = SetptA
SPBOut = SetptB

Unoccupied setpoint crossover is prevented by comparing UnocSPB and UnocSPA values.

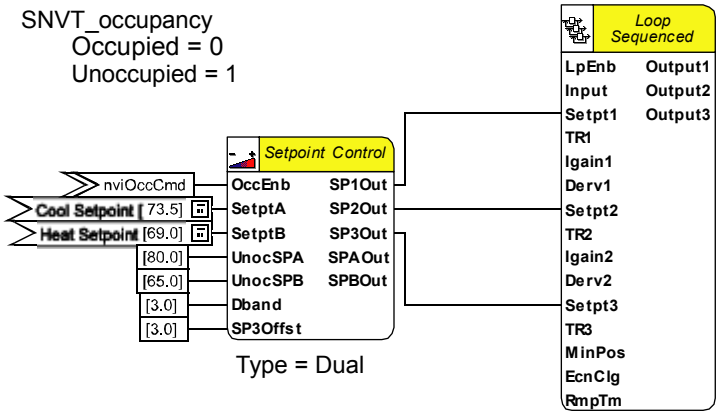
If UnocSPB > UnocSPA, the UnocSPB = UnocSPA.

A not active (NA) value at these object inputs produces these results:

- If SetptA is NA, SPAOut is set to NA.
- If SetptB is NA, SPBOut is set to NA.
- If UnoccSPA is NA, SP1Out and SP3Out are set to NA.
- If UnoccSPB is NA, SP2Out is set to NA.
- If Dband is a negative value or NA, the value 0.0 is used as Dband.
- If SP3Offst is a negative value or NA, the value 0.0 is used as SP3Offst.

Example

Figure–5.162 illustrates a Setpoint Control object used to provide the control setpoints for a heating, cooling, and economizer control sequence. In this example, the occupied and unoccupied mode of operation is received as a variable from the network. The heating and cooling setpoint values are established using an MN-Sx wall sensor and setpoint device.



Figure–5.162 Example Setpoint Control Object.

SqRt Mul / Add

Object Usage: The SqRt Mul / Add object is a three-input math object for use with analog values (AV). This object takes the square root of Input[1] and multiplies it with Input[2]. Input[3] is then added to the multiplication result.

Output = [(SQRT AV1) x AV2] + AV3

A valid input value is required on Input[1] for the object to operate.

Device Support: (all firmware revisions)

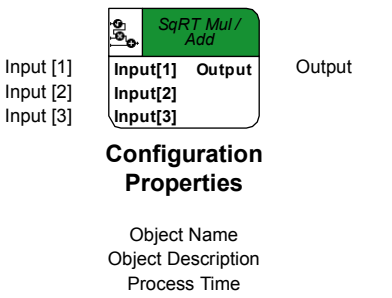
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.288 SqRt Mul / Add Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.289 SqRt Mul / Add Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The square root of this value is the multiplicand in the equation. [(SQRT **AV1**) x AV2] + AV3	-163.83 to 16383	If not active (NA) is present, the output is set to NA.
Input[2]	Input [2]	Class: Analog - The second value used as the multiplier to the first value square root result. [(SQRT AV1) x **AV2**] + AV3	-163.83 to 16383	If not active (NA), the term is ignored.
Input[3]	Input [3]	Class: Analog - The third value added to the previous result. [(SQRT AV1) x AV2] + **AV3**	-163.83 to 16383	If not active (NA), the term is ignored.

Table–5.290 SqRt Mul / Add Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output is the result of the equation: [(SQRT AV1) x AV2] + AV3. Refer to Truth Table for effects of input(s) with not active (NA).	-163.83 to 16383

Applying the Object

The SqRt Mul / Add object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. This object is the only math object with a square root operator. The equation specific to the SqRt Mul / Add object is:

[(SQRT AV1) x AV2] + AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. Table–5.291 shows how NA inputs affect the SqRt Mul / Add object output.

Table–5.291 SqRt Mul / Add Object Truth Table.

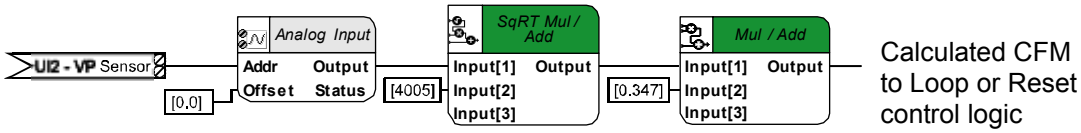
Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	[(SQRT AV1) x AV2] + AV3
AV1	AV2	NA	(SQRT AV1) x AV2
AV1	NA	AV3	(SQRT AV1) + AV3
AV1	NA	NA	(SQRT AV1)
Negative	AV2	AV3	[(0) x AV2] + AV3
NA	AV2	AV3	NA
NA	NA	NA	NA
NA	NA	AV3	NA

Example Application

In this example, the SqRt Mul / Add object is used in the conversion of a duct velocity pressure (Pv) value to a CFM airflow value (along with another math object, the Mul / Add object). The formula used is:

$$CFM = \sqrt{Pv} \times (4005) \times (\text{duct area in square feet})$$

The velocity pressure sensor is installed in an 8” duct, which has an area of 0.347 square feet. The two math objects appear below Figure-5.163.



Figure–5.163 SqRt Mul / Add Object Used in CFM Calculation.

SR Flip-Flop

Object Usage: The SR Flip-Flop is a two-input logic object for use with digital values (DV). It performs a set-reset flip-flop logic function. In operation, the SR Flip-Flop is similar to the three-input Clocked SR object (page 186). However, the SR Flip-Flop object does not contain the additional Clock input that requires a coinciding OFF/ON/OFF transition (clock) before the Output changes. Output changes are based on input transition at the Set and Reset inputs. An unconnected input is considered invalid or not active (NA), and is ignored in the object's algorithm. If both inputs are NA, the output is also set to NA.

Output = Set-Reset Flip Flop

Device Support: (all firmware revisions)

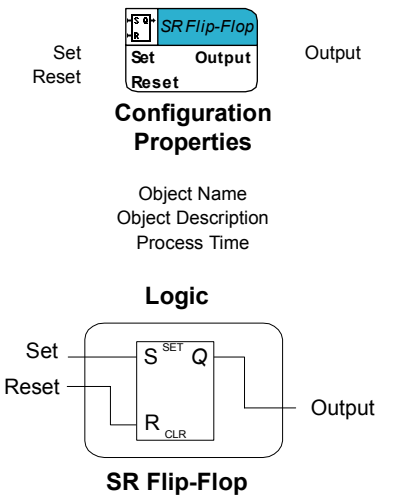
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 8 bytes
RAM: 10 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Logic and Math Control

Reference Listing of All Digital Logic Objects

Object Name	Digital Object Algorithm (all are three-input unless noted)
AND / AND	In1 AND In2 AND In3
AND / OR	(In1 AND In2) OR In3
Clocked SR	Clocked Set-Reset Flip-Flop Logic
EXOR	Two-input, Exclusive OR
Latch	Digital Sample and Hold or Latch
OR / AND	(In1 OR In2) AND In3
OR / OR	In1 OR In2 OR In3
SR Flip-Flop	Two-input, Set-Reset Flip-Flop Logic

Properties

Table–5.292 SR Flip-Flop Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.293 SR Flip-Flop Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Set	Set	Class: Digital - Evaluated as the set input. A not active (NA) is disregarded by the SR Flip Flop algorithm.	—	See the Truth Table and Timing Diagram for input to output sequence of operation.
Reset	Reset	Class: Digital - Evaluated as the reset input. A not active (NA) is disregarded by the SR Flip Flop algorithm.	—	See the Truth Table and Timing Diagram for input to output sequence of operation.

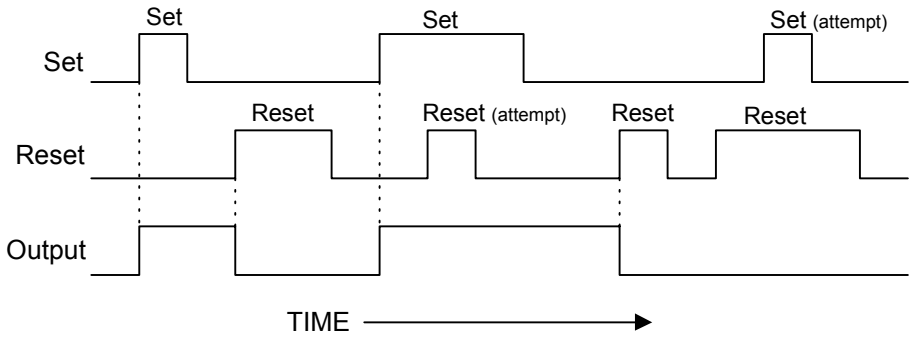
Table–5.294 SR Flip-Flop Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Digital - The output indicates the result of the SR Flip Flop logic algorithm. If not active (NA) is present at both inputs, the output is set to NA.	OFF (0.0) ON (100.0)

Applying the Object

The SR Flip-Flop is a two-input logic object that processes digital values (DV) and produces a digital output result. It is most similar to the three-input Clocked SR object. The SR Flip-Flop object toggles (flip-flop) its object Output in concert with DV changes on the inputs Set and Reset.

A timing diagram Figure-5.164 and truth table Table–5.295 help illustrate the operation of the SR Flip-Flop.



Figure–5.164 Timing Diagram for an SR Flip-Flop Object.

Table–5.295 Truth Table for SR Flip-Flop.

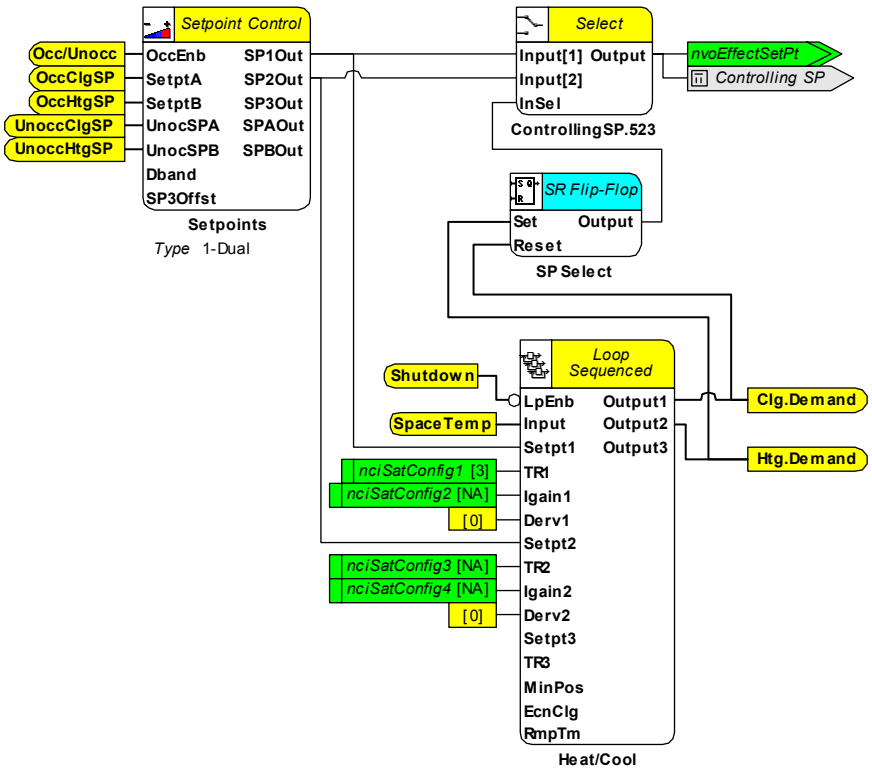
	Set	Reset	Output
Hold	OFF	OFF	No change
Set	ON	OFF	ON
Reset	OFF	ON	OFF
Prohibited	ON	ON	No change

Note: Following a controller reset the object Output is NA until an object input forces Set or Reset to ON or OFF.

As with other logic objects, Inputs to this object are typically digital values from a connection to another object's output, represented numerically with a 0 (OFF) or 100 (ON). However, Inputs can also process analog values or a value from an assigned constant. An Input with an analog value or constant is evaluated as OFF if the value is negative or equal to zero (0.0), or ON if the Input has any positive value greater than zero.

Example

The SR Flip-Flop object below Figure-5.165 is used like one in several of the Invensys pre-engineered applications for MN 100 and 200 controllers. The object output drives the selection of the currently active (effective) setpoint, based on the loop outputs of the Loop Sequenced object.



Figure–5.165 SR Flip-Flop Object as Used in Pre-Engineered Application.

Step Driver

Object Usage: The Step Driver object provides a method of sequencing up to 6 digital outputs based on a proportional 0.0 to 100.0% signal received at the object Input. The object is similar to the Sequence (6) object (page 481), but differs by providing independent adjustment of each output stage via separate Trip Point and Deadband inputs. Because each output stage is evaluated separately from the other stages, custom or nonlinear output sequences are possible. The Stages On output indicates the number of output stages currently ON.

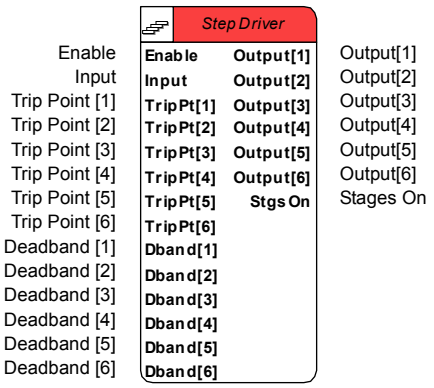
Device Support:

MN 800 series

Memory Requirements: (per object)

EEPROM: 32 bytes
RAM: 14 bytes

WP Tech Representation



Configuration Properties

Object Name
Object Description

WP Tech Stencil:
Timer and Sequence Control

Properties

Table–5.296 Step Driver Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.

Table–5.297 Step Driver Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Enable	Enable	Class: Digital - An ON or not active (NA) enables the Step Driver object algorithm. An OFF causes Outputs[1] through [6] to be reset and held in an OFF state, and the Stages On output to be set to 0 (zero).	—	
Input	Input	Class: Analog - The monitored value (0.0 to 100.0%) that determines the output action, as configured by the values at the Trip Point and Deadband inputs. A not active (NA) is evaluated the same as 0.0%.	0.0 to 100.0%	Negative value defaults to 0.0% and a value greater than 100.0% defaults to 100.0%.

Table–5.297 Step Driver Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
TripPt[1]	Trip Point [1]	Class: Analog - Defines the “ON” trip point for Output[1]. The output is set to ON whenever the input value is greater than or equal to this Trip Point.	0.0 to 100.0%	A Trip Point value of zero causes the associated output to be held in the Digital OFF condition. Any Trip Point of not active (NA) causes that output to be reset and held in an OFF state. Negative values default to 0.0% and values greater than 100.0% default to 100.0%.
TripPt[2]	Trip Point [2]	Class: Analog - Defines the “ON” trip point for Output[2]. The output is set to ON whenever the input value is greater than or equal to this Trip Point.	0.0 to 100.0%	
⋮	⋮	⋮	⋮	
TripPt[6]	Trip Point [6]	Class: Analog - Defines the “ON” trip point for Output[6]. The output is set to ON whenever the input value is greater than or equal to this Trip Point.	0.0 to 100.0%	
Dband[1]	Deadband [1]	Class: Analog - This value is used to determine the “OFF” trip point for Output[1]. The output is set to OFF whenever the input value is less than the result of subtracting the Deadband[1] value from the Trip Point[1] value.	0.0 to 100.0%	Any Deadband of not active (NA) internally defaults to a deadband of 0.0%. Negative values default to 0.0% and values greater than 100.0% default to 100.0%.
Dband[2]	Deadband [2]	Class: Analog - This value is used to determine the “OFF” trip point for Output[2]. The output is set to OFF whenever the input value is less than the result of subtracting the Deadband[2] value from the Trip Point[2] value.	0.0 to 100.0%	
⋮	⋮	⋮	⋮	
Dband[6]	Deadband [6]	Class: Analog - This value is used to determine the “OFF” trip point for Output[6]. The output is set to OFF whenever the input value is less than the result of subtracting the Deadband[6] value from the Trip Point[6] value.	0.0 to 100.0%	

Table–5.298 Step Driver Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output[1]	Output [1]	Class: Digital - This output is set to ON whenever the input value is greater or equal to the Trip Point[1] value. The output remains ON until the input drops below the result of subtracting Deadband[1] from Trip Point[1].	OFF (0.0) ON (100.0)
Output[2]	Output [2]	Class: Digital - This output is set to ON whenever the input value is greater or equal to the Trip Point[2] value. The output remains ON until the input drops below the result of subtracting Deadband[2] from Trip Point[2].	OFF (0.0) ON (100.0)
⋮	⋮	⋮	⋮
Output[6]	Output [6]	Class: Digital - This output is set to ON whenever the input value is greater or equal to the Trip Point[6] value. The output remains ON until the input drops below the result of subtracting Deadband[6] from Trip Point[6].	OFF (0.0) ON (100.0)
StgsOn	Stages On	Class: Analog - Reflects the actual number of output stages ON during active step driver operation.	0, 1, 2, 3, 4, 5, or 6

Applying the Object

The Step Driver object provides a means of generating a digital output sequence, based upon programmable trip points. Each stage is evaluated independently of the other stages, allowing for unique linear and nonlinear sequences. Each stage's trip point and deadband combination is compared to the input value, and the digital output is set or reset accordingly.

Enable: The Enable input controls the active operation of the Step Driver object. An Enable value of digital OFF disables the Step Driver object, causing all outputs to be reset and held in the digital OFF state, and the Stages On output is set to a value of 0 (zero). An ON or not active (NA) at the Enable input enables the Step Driver object, allowing outputs to operate based upon the Input value.

Input: The Step Driver control algorithm calculates the digital output states based upon the Input property's value. The values applied to the Input property are limited to a range between 0.0 and 100.0%. An Input value of not active (NA) is evaluated as a 0.0% value.

Trip Point [1] through [6]: Each Trip Point input value determines that stage's "ON" trip point. The "ON" trip point is the Input property value at which the associated output is set to ON. The associated output (Output[1] through Output[6]) is set to digital ON whenever the Input property value is greater than or equal to the associated Trip Point value.

Output[x] = ON when Input $\geq$ Trip Point[x]

Each Trip Point can be assigned a value between 0.0 and 100.0%. A Trip Point value of zero causes the associated output to be held in the Digital OFF condition. Trip Point values above and below this range are limited internally. A Trip Point value of not active (NA) causes the individual stage's comparison to be ignored, whereby the output is reset and held in the OFF state.

Deadband [1] through [6]: Each Deadband value is used to determine that stage's "OFF" trip point. This is the point at which the associated output (Output[1] through Output[6]) is set to OFF, which occurs whenever the Input properties value is less than the difference between the corresponding Trip Point and Deadband values.

Output[x] = OFF when Input < Trip Point[x] - Deadband[x]

Each Deadband can be set to a value between 0.0 and 100.0%. Deadband values above and below this range are limited internally. A Deadband of not active (NA) causes the stage's Deadband to default to a value of 0.0%.

Each stage's calculated "OFF" trip point ([Trip Point] - [Deadband]) must be a positive value. Any OFF trip point determined to be negative (< 0.0%) is evaluated as 0.0%, which guarantees that all outputs will be reset to OFF whenever the Input value is 0.0%.

Reset Operation

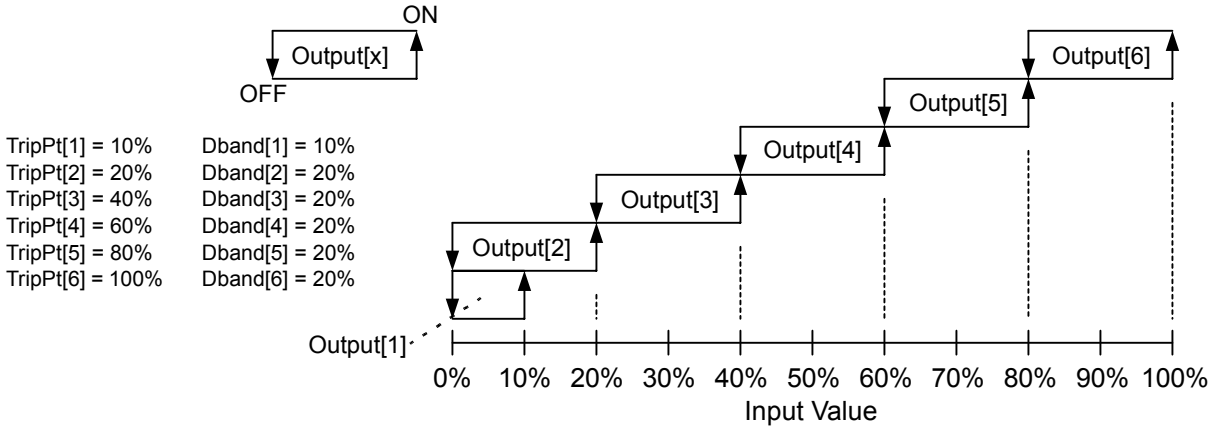
Reset of the object (controller reset) causes all outputs to be initialized OFF and the Stages On output to be initialized to a value of 0 (zero). The object then activates the required outputs based on the current input values, and the Stages On output indicates the actual number of outputs that are ON.

Examples

Two Step Driver object examples include a linear control sequence configuration and a nonlinear control sequence configuration.

Linear Configuration

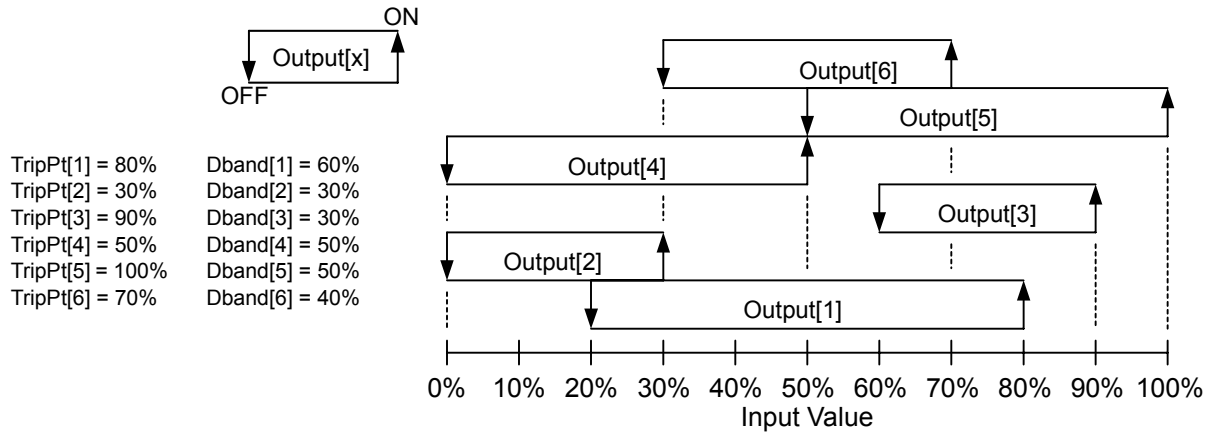
A Step Driver object is configured to generate a linear sequence shown in Figure–5.166. Output[1] comes ON before Output[2], and remains ON until Output[2] returns OFF, and so on, through the full range of the outputs.



Figure–5.166 Step Driver Object Operation in Linear Sequence Configuration.

Nonlinear Configuration

A Step Driver object is configured to generate a nonlinear sequence shown in Figure–5.167. In this example, the variations in trip ON and OFF points provide an irregular output response.



Figure–5.167 Step Driver Object Operation in Linear Sequence Configuration.

Sub / Add

Object Usage: The Sub / Add object is a three-input math object for use with analog values (AV). This object subtracts Input[2] from Input[1] and then adds Input[3] to the result.

Output = (AV1 - AV2) + AV3

A minimum of two valid inputs are required to produce a valid output.

Device Support: (all firmware revisions)

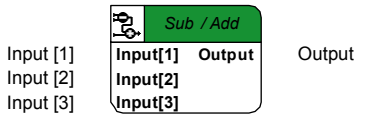
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.299 Sub / Add Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.300 Sub / Add Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The first value or minuend in the equation: (**AV1** - AV2) + AV3	-163.83 to 16383	If not active (NA) is present at both Inputs[1] and [2], the output is set to NA.
Input[2]	Input [2]	Class: Analog - The second value or subtrahend, subtracted from the first value: (AV1- **AV2**) + AV3	-163.83 to 16383	
Input[3]	Input [3]	Class: Analog - The third value added to the previous result. (AV1 - AV2) + **AV3**	-163.83 to 16383	If not active (NA), the output is set to NA.

Table–5.301 Sub / Add Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog -The output is the result of the equation: (AV1 - AV2) + **AV3** Refer to Truth Table for effects of input(s) with not active (NA).	-163.83 to 16383

Applying the Object

The Sub / Add object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. The equation specific to the Sub / Add object is:

(AV1 - AV2) + AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. Table–5.302 shows how NA inputs affect the output of the Sub / Add object.

Table–5.302 Sub / Add Object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	(AV1 - AV2) + AV3
AV1	AV2	NA	NA
AV1	NA	AV3	AV1 + AV3
NA	AV2	AV3	-AV2 + AV3
NA	NA	AV3	NA
NA	NA	NA	NA

Sub / Div

Object Usage: The Sub / Div object is a three-input math object for use with analog values (AV). This object subtracts Input[2] from Input[1] and then divides the result by Input[3].

Output = (AV1 - AV2) ÷ AV3

A minimum of two valid inputs are required to produce a valid output.

Device Support: (all firmware revisions)

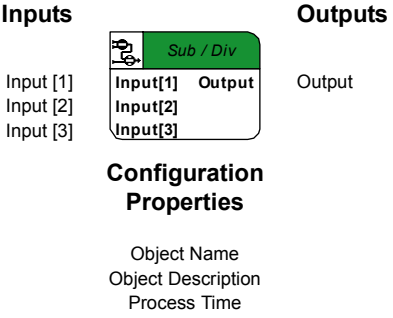
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.303 Sub / Div Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.304 Sub / Div Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The first value or minuend in the equation: (**AV1** - AV2) ÷ AV3	-163.83 to 16383	If not active (NA) is present at both Inputs[1] and [2], the output is set to NA.
Input[2]	Input [2]	Class: Analog - The second value or subtrahend, subtracted from the first value: (AV1 - **AV2**) ÷ AV3	-163.83 to 16383	
Input[3]	Input [3]	Class: Analog - The third value or divisor, which is divided into the previous result: (AV1 - AV2) ÷ **AV3**	-163.83 to 16383	If not active (NA), the output is set to NA.

Table–5.305 Sub / Div Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output is the result of the equation: (AV1 - AV2) ÷ AV3. Refer to Truth Table for effects of input(s) with not active (NA).	-163.83 to 16383

Applying the Object

The Sub / Div object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. The equation specific to the Sub / Div object is:

(AV1 - AV2) ÷ AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Note: A value of zero at Input[3] causes the output to process a “divide by zero” that sets the output to either a minimum (-163.83) or a maximum (16383) value based upon the results of the first two inputs. A negative result causes the output to be set to the minimum (-163.83) value. A positive result causes the output to be set to the maximum (16383) value.

- Result < 0 sets the output to the minimum (-163.83) value.
- Result ≥ 0 sets the output to the maximum (16383) value.

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. Table–5.306 shows how NA inputs affect the output of the Sub / Div object.

Table–5.306 Sub / Div Object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	(AV1 - AV2) ÷ AV3
AV1	AV2	NA	NA
AV1	NA	AV3	AV1 ÷ AV3
NA	AV2	AV3	-AV2 ÷ AV3
NA	NA	AV3	NA
NA	NA	NA	NA

Sub / Mul

Object Usage: The Sub / Mul object is a three-input math object for use with analog values (AV). This object subtracts Input[2] from Input[1] and then multiplies the result with Input[3].

Output = (AV1 - AV2) x AV3

A minimum of two valid inputs are required to produce a valid output.

Device Support: (all firmware revisions)

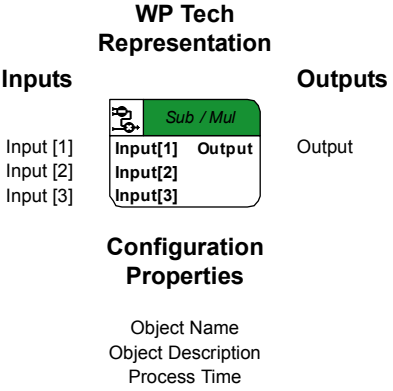
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.307 Sub / Mul Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.308 Sub / Mul Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog - The first value or minuend in the equation: (**AV1** - AV2) x AV3	-163.83 to 16383	If not active (NA) is present at both Inputs[1] and [2], the output is set to NA.
Input[2]	Input [2]	Class: Analog - The second value or subtrahend, subtracted from the first value: (AV1 - **AV2**) x AV3	-163.83 to 16383	
Input[3]	Input [3]	Class: Analog - The third value multiplied to the previous result. (AV1 - AV2) x **AV3**	-163.83 to 16383	If not active (NA), the output is set to NA.

Table–5.309 Sub / Mul Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output is the result of the equation: (AV1 - AV2) x AV3. Refer to Truth Table for effects of input(s) with not active (NA).	-163.83 to 16383

Applying the Object

The Sub / Mul object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. The equation specific to the Sub / Mul object is:

(AV1 - AV2) x AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. Table–5.310 shows how NA inputs affect the output of the Sub / Mul object.

Table–5.310 Sub / Mul Object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	(AV1 - AV2) x AV3
AV1	AV2	NA	NA
AV1	NA	AV3	AV1 x AV3
NA	AV2	AV3	-AV2 x AV3
NA	NA	AV3	NA
NA	NA	NA	NA

Sub / Sub

Object Usage: The Sub / Sub object is a three-input math object for use with analog values (AV). This object subtracts Input[2] from Input[1] and then subtracts Input[3] from the result.

Output = (AV1 - AV2) - AV3

Only one valid input is required to produce a valid output.

Device Support: (all firmware revisions)

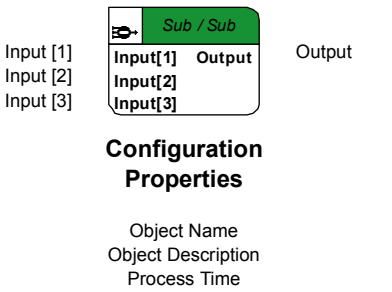
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 12 bytes (standard controller)
2 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Logic and Math Control

Reference Listing of All Three-input Math Objects

Object Name	Object Algorithm
Abs Sub / Div	| AV1 - AV2 | ÷ AV3
Add / Add	AV1 + AV2 + AV3
Add / Div	(AV1 + AV2) ÷ AV3
Average	Average (AV1, AV2, AV3)
MA Volume	RA - ([(RA - OA) x MASetpt] ÷ 100)
Mul / Add	(AV1 x AV2) + AV3
Mul / Div	(AV1 x AV2) ÷ AV3
SqRt Mul / Add	[(SQRT AV1) x AV2] + AV3
Sub / Add	(AV1 - AV2) + AV3
Sub / Div	(AV1 - AV2) ÷ AV3
Sub / Mul	(AV1 - AV2) x AV3
Sub / Sub	(AV1 - AV2) - AV3

Properties

Table–5.311 Sub / Sub Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.312 Sub / Sub Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
Input[1]	Input [1]	Class: Analog -The first value or minuend in the equation. (**AV1** - AV2) - AV3	-163.83 to 16383	If not active (NA), this input is ignored.
Input[2]	Input [2]	Class: Analog - The second value or first subtrahend, subtracted from the first value. (AV1 - **AV2**) - AV3	-163.83 to 16383	If not active (NA), this input is ignored.
Input[3]	Input [3]	Class: Analog - The third value or second subtrahend, subtracted from the previous result. (AV1 - AV2) - **AV3**	-163.83 to 16383	If not active (NA), this input is ignored.

Table–5.313 Sub / Sub Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Output	Output	Class: Analog - The output is the result of the equation: (AV1 -AV2) - AV3 . This output is not active (NA) only if all three inputs are NA.	-163.83 to 16383

Applying the Object

The Sub / Sub object is similar to other three-input math objects, which also process analog values (AV) and produce an AV output. The equation specific to the Sub / Sub object is:

(AV1 - AV2) - AV3

As with other math objects, inputs to this object are typically analog values, but may also be numerical representations of digital values (0.0 for OFF or 100.0 for ON), or not active (NA).

Not Active Inputs

If unconnected, an input is considered not active (NA). An analog value received on a connected input from another object may also change from a valid value to NA, depending on the behavior of the sending object. Table–5.314 shows how NA inputs affect the output of the Sub / Sub object.

Table–5.314 Sub / Sub Object Truth Table.

Input[1]	Input[2]	Input[3]	Output
AV1	AV2	AV3	(AV1 - AV2) - AV3
AV1	AV2	NA	AV1 - AV2
AV1	NA	NA	AV1
NA	AV2	AV3	-AV2 - AV3
NA	AV2	NA	-AV2
NA	NA	AV3	-AV3
NA	NA	NA	NA

Thermostat

Object Usage: The Thermostat object provides two-position ON / OFF output control with hysteresis (differential). Both a direct acting and a reverse acting output are provided. Applications include typical cooling type or heating type control sequences, or any sequence where two-position control based on setpoint differential is required. Outputs are set to not active (NA) when any of the three inputs are NA.

Note: Controllers with Rev.3 or later firmware also have another closely-related object available, the Thermostat 2 object (page 522). That object offers all functions of the Thermostat object plus additional capability.

Device Support: (all firmware revisions)

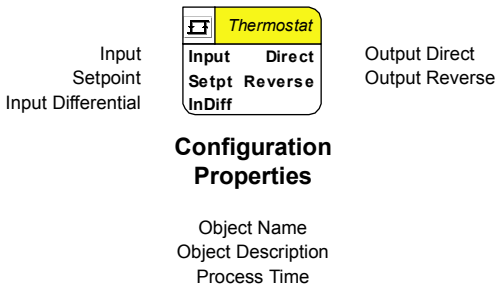
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F1, F2, F3, H1, H2, H3, R1, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 10 bytes
RAM: 14 bytes (standard controller)
4 bytes (MN 800)

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
Loop and Process Control

Properties

Table–5.315 Thermostat Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.316 Thermostat Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input	Input	Class: Analog -The control signal input for the object.	-163.83 to 16383	A not active (NA) sets both the direct and reverse outputs to NA.
Setpt	Setpoint	Class: Analog - The control setpoint used by the object.	-163.83 to 16383	A not active (NA) sets both the direct and reverse outputs to NA.
InDiff	Input Differential	Class: Analog - Defines the differential used to calculate the ON / OFF control points for both the direct and reverse acting outputs.	0 to 16383	A not active (NA) sets both the direct and reverse outputs to NA.

Table–5.317 Thermostat Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Direct	Output Direct	Class: Digital - The direct acting output, this output is set to ON whenever the Input value rises to or above the Setpoint plus one half the Input Differential. This output remains ON until the Input drops to or below the Setpoint minus one half the Input Differential, whereby it goes OFF. This output is set to not active (NA) when any of the three inputs is NA.	OFF (0.0) ON (100.0)
Reverse	Output Reverse	Class: Digital - The reverse acting (heating) output, this output is set to ON whenever the Input value drops to or below the Setpoint minus one half the Input Differential. This output remains ON until the Input rises to or above the Setpoint plus one half the Input Differential, whereby it goes OFF. This output is set to not active (NA) when any of the three inputs is NA.	OFF (0.0) ON (100.0)

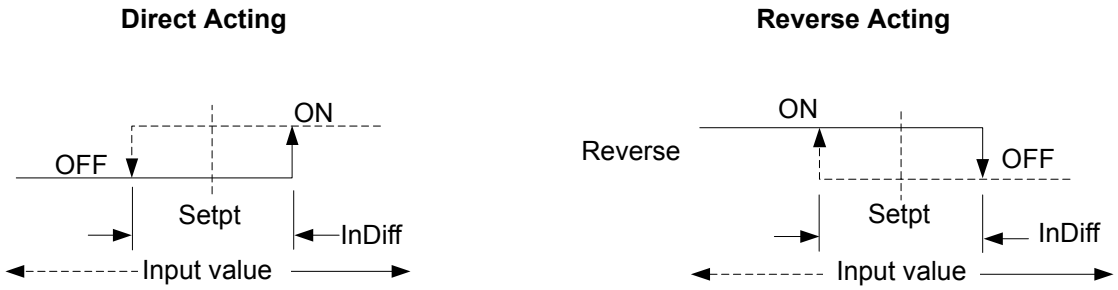
Applying the Object

The Setpoint Control object is typically used for simple ON / OFF control based on a setpoint differential, such as thermostatic control in a cooling or heating application. The two available outputs (Output Direct and Output Reverse) operate inversely, with each corresponding to a either a direct or reverse control action.

During operation, the differential is always symmetric to the setpoint value. No change occurs at the outputs when the input signal is within the differential span surrounding the setpoint.

Direct Acting and Reverse Acting

Upon a controller reset, both the direct and reverse outputs are initialized to an OFF condition. The object outputs Output Direct (Direct) and Output Reverse (Reverse) are then set to ON or OFF based on the values received at the Input Differential (InDiff), Input, and Setpoint (Setpt) as shown in Figure–5.168.



Figure–5.168 Thermostat Object Direct and Reverse Output Action.

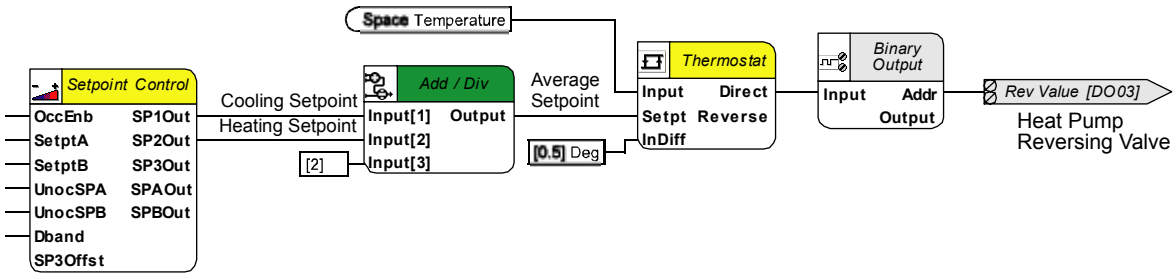
The direct acting output is set to ON whenever the Input rises to or above the Setpoint plus one half the Input Differential. The output remains ON until the input drops to or below the Setpoint minus one half the Input Differential. The output then returns to an OFF condition. This output is set to not active (NA) whenever any of the three inputs are NA.

The reverse acting output is set to ON whenever the Input drops to or below the Setpoint minus one half the Input Differential. The output remains ON until the input rises to or above the Setpoint plus one half the Input Differential. The output then returns to an OFF condition. This output is set to not active (NA) whenever any of the three inputs are NA.

Note: If a reset occurs when either the Direct or Reverse output is ON but the Input value is within the Input Differential, that output will remain OFF until the Input value again either rises to or above (or drops to or below) the Setpoint plus (or minus) one half of the Input Differential.

Example

Figure–5.169 shows a Thermostat object used to switch a reversing valve in a heat pump application. The Thermostat object monitors the space temperature and compares it to a setpoint that is the average of the current cooling setpoint and heating setpoint (supplied by the Add / Div object). An Input Differential of 0.5 degrees and the Output Direct output is used for reversing valve control.



Figure–5.169 Example Thermostat Object.

Thermostat 2

(Rev.3 or Later Firmware Required)

Object Usage: Similar to the Thermostat object (page 519), the Thermostat 2 object provides two-position ON / OFF output control with hysteresis (differential). The Thermostat 2 object differs by offering an additional Setpoint Reference input. If desired, this permits the adjustment of the trip points (On and Off) defined by the Input Differential to be done “off center” from the setpoint.

Note: If the Setpoint Reference input is left unconnected or has a not active (NA), the object functions identically to the Thermostat object.

Both direct acting and reverse acting outputs are provided. Applications include typical cooling type or heating type control sequences, or any sequence where two-position control based on setpoint differential is required. Both outputs are set to NA whenever any of the three main inputs (Input, Setpoint, Input Differential) are NA.

Device Support: (See page 7)

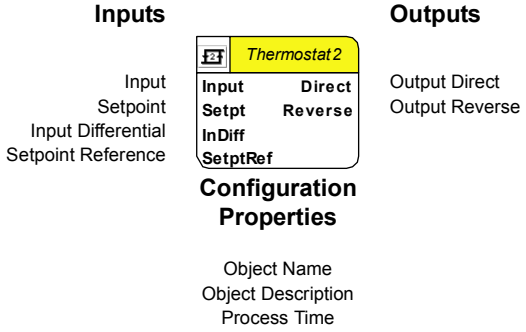
MNL-5Rxx, -10Rxx, -15Rxx, -20Rxx,
where xx = F2, F3, H2, H3, R2, R3, S1, S2, S3, or S4
MNL-11Rxx, -13Rxx
where xx = F2 or F3
MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V2 or V3

MN 800 series

Memory Requirements: (per object)

EEPROM: 12 bytes
RAM: 16 bytes (standard controller)
4 bytes (MN 800)

WP Tech Representation



WP Tech Stencil:
Loop and Process Control

Properties

Table–5.318 Thermostat 2 Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.

Table–5.318 Thermostat 2 Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.319 Thermostat 2 Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selections	Notes
Input	Input	Class: Analog -The control signal input for the object.	-163.83 to 16383	A not active (NA) sets both the direct and reverse outputs to NA.
Setpt	Setpoint	Class: Analog - The control setpoint used by the object.	-163.83 to 16383	A not active (NA) sets both the direct and reverse outputs to NA.
InDiff	Input Differential	Class: Analog - Defines the differential used to calculate the ON / OFF control points for both the direct and reverse acting outputs.	0 to 16383	A not active (NA) sets both the direct and reverse outputs to NA.
SetptRef	Setpoint Reference	Class: Analog - Defines the reference point used to calculate the direct and reverse acting output trip points in reference to the setpoint value. A setpoint reference of 50% “centers” the trip points around the setpoint.	0 to 100%	A not active (NA) or unconnected SetptRef results in a default Setpoint Reference of 50%. This causes the object to function like the Thermostat object.

Table–5.320 Thermostat 2 Object Output Properties.

Abbrev.	Name	Class / Description	Valid Values
Direct	Output Direct	Class: Digital - The direct acting output, this output is set to ON whenever the Input value rises to or above the calculated On trip point. This output remains ON until the Input drops to or below the calculated Off trip point, whereby it goes OFF. This output is set to not active (NA) when any of the three main inputs (Input, Setpoint, Input Differential) is NA.	OFF (0.0) ON (100.0)
Reverse	Output Reverse	Class: Digital - The reverse acting output, this output is set to ON whenever the Input value drops to or below the calculated On trip point. This output remains ON until the Input rises to or above the calculated Off trip point, whereby it goes OFF. This output is set to not active (NA) when any of the three main inputs (Input, Setpoint, Input Differential) is NA.	OFF (0.0) ON (100.0)

Applying the Object

The Thermostat 2 object can be used as a direct replacement of the Thermostat object. Additionally, the Thermostat 2 object allows adjustment as needed in the trip ON and trip OFF points around the setpoint. Unlike the Thermostat object, the differential does not have to be symmetric around the setpoint value. In all other aspects, the Thermostat 2 object operates like the Thermostat object. No change occurs at the outputs when the input signal is within the differential span surrounding the setpoint.

Thermostat 2 objects are typically used for simple ON / OFF control based on a setpoint differential, such as thermostatic control in a cooling or heating application. The two available outputs (Output Direct and Output Reverse) operate inversely, with each corresponding to a either a direct or reverse control action.

Direct Acting and Reverse Acting

Upon a controller reset, both the direct and reverse outputs are initialized to an OFF condition. The outputs Output Direct (Direct) and Output Reverse (Reverse) are then set to ON or OFF based on the values received at the Input Differential (InDiff), Input, Setpoint (Setpt), and Setpoint Reference (SetptRef) using the following calculations:

Direct Acting Trip Points

Output Direct ON when:Input ≥ (Setpoint + [(100% - SetptRef%) x InDiff])

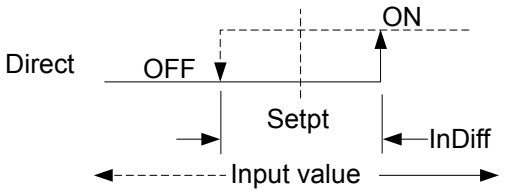
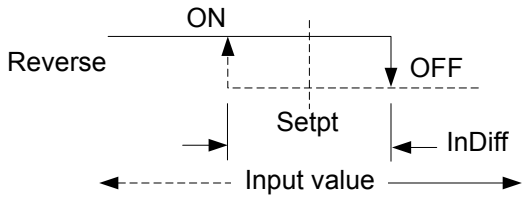
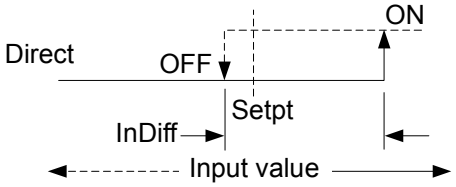
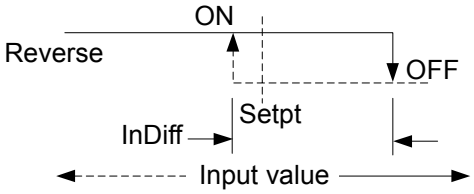
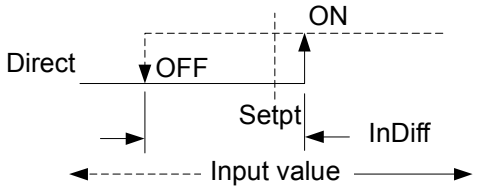
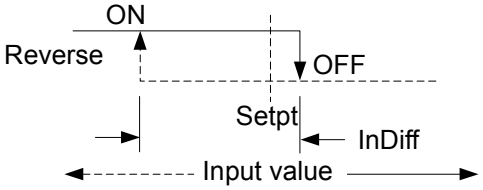
OFF when: Input ≤ (Setpoint - [SetptRef% x InDiff)])

Reverse Acting Trip Points

Output Reverse ON when: Input ≤ (Setpoint - [SetptRef% x InDiff)])

OFF when: Input ≥ (Setpoint + [(100% - SetptRef%) x InDiff])

Figure–5.170 shows the Direct Acting and Reverse Acting output action at three different levels of Setpoint Reference: 50%, 20%, and 80%.

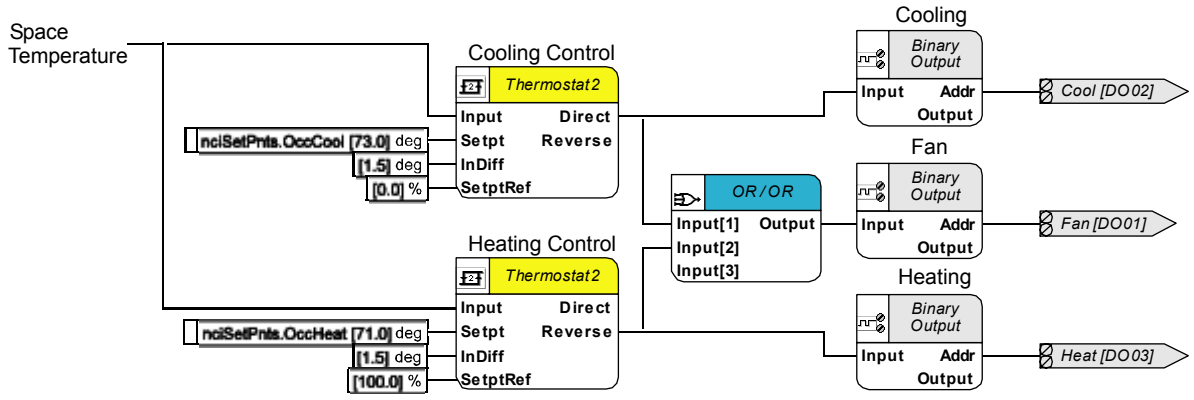
Setpoint Reference	Output Direct	Output Reverse
50% (or NA)		
20%		
80%		

Figure–5.170 Thermostat 2 Object Direct and Reverse Output Action (3 Different Setpoint Reference Levels).

Note: If a reset occurs when either the Direct or Reverse output is ON but the Input value is within the Input Differential, that output will remain OFF until the Input value again either rises to or above (or drops to or below) the necessary Trip ON point.

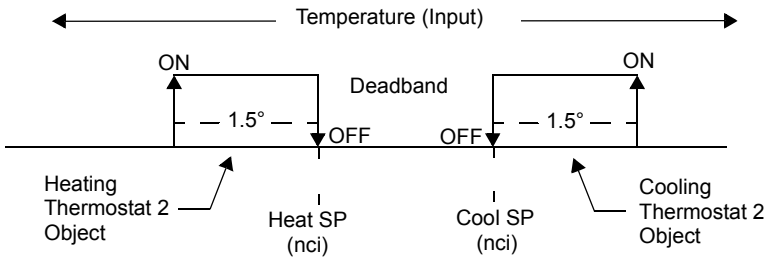
Example

This example application uses two Themostat 2 objects, one for cooling control and one for heating. A fan is activated whenever either connected Thermostat 2 output is ON.



Figure–5.171 Simple Thermostat 2 Object Application Example.

The cooling and heating Thermostat 2 object setpoint reference properties are set to 0% and 100%, respectively. This ensures a control deadband between the cooling and heating setpoints, as shown in Figure–5.172 below.



Figure–5.172 Example Thermostat 2 Object Actions.

In this example, both setpoints are sourced from an NCI in the controller's profile, nciSetPnts.

Note: This example application has been simplified and does not include possible additional logic for "control safeties".

VAV Actuator

Object Usage: The VAV Actuator object is a point-type object used to operate the VAV terminal box damper using either a MicroNet VAV controller with integral actuator (MNL-V1 or V2) or a Micronet VAV controller (MNL-V3) connected to an external floating-control type actuator. The VAV Actuator object works in conjunction with the Pressure Transducer object (page 406), another object specific to MicroNet VAV controllers. That object supplies airflow feedback and a calibration factor to specific inputs of the VAV Actuator object. The VAV Actuator object activates its Drive outputs when differences occur between the received feedback signal and the airflow setpoint (received from the VAV application). Other object inputs include actuator Drive Time, Force Open / Force Close, and an Action input that can reverse actuator rotation (output drive sequence).

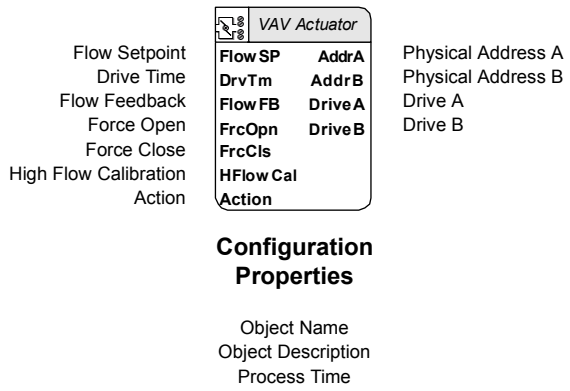
Device Support: (all firmware revisions)

MNL-V1Rxx, -V2Rxx, -V3Rxx, where xx = V1, V2, or V3

Memory Requirements: (per object)

EEPROM: 22 bytes RAM: 40 bytes

WP Tech Representation



Configuration Properties

Object Name
Object Description
Process Time

WP Tech Stencil:
IO and Alarm Control

Properties

Table–5.321 VAV Actuator Object Configuration Properties.

Abbrev.	Name	Class / Description	Default	Range / Selection	Notes
Name	Object Name	Class: Character String - The user-defined name for the object, unique within the controller where the object resides.	—	—	Printable characters only. See Object Name on page 89 for more details.
Desc	Description	Class: Character String - Optional user-defined descriptor available to further describe the object.	—	—	Stored in the WPT file only. See Object Description on page 89 for more details.
ProTm	Process Time	Class: Analog - Defines the frequency at which the object executes its algorithm.	4	6 - Low 4 - Medium 2 - High	See Process Time on page 90 for more details.

Table–5.322 VAV Actuator Object Input Properties.

Abbrev.	Name	Class / Description	Range / Selection	Notes
FlowSP	Flow Setpoint	Class: Analog - The requested airflow control point (CFM or liters per second) compared against the Flow Feedback value. The object algorithm uses both to determine the physical output and Drive output actions.		A negative value or not active (NA) causes object outputs to hold at their current conditions.
DrvTm	Drive Time	Class: Analog - Defines the actuator full-stroke drive time in seconds. A not active (NA) or value less than 10 seconds causes the object to use an internal default Drive Time specific to the integral actuator for the MicroNet VAV controllers.	10 to 1000 seconds	Typically left unconnected (NA) if for an integral actuator.
FlowFB	Flow Feedback	Class: Analog - The appropriate airflow feedback signal representing the current measured variable air volume flow rate (CFM or liters per second). This value is typically supplied by the VAV Pressure Transducer object, and is compared to the Flow Setpoint input value.	—	A negative value or not active (NA) indicates loss of valid feedback and causes object outputs to hold at their current conditions.
FrcOpn	Force Open	Class: Digital - An ON causes the actuator to drive towards the full open condition for a time period equal to the full-stroke Drive Time assigned. An OFF or not active (NA) disables the Force Open request.		These inputs remain functional even if other inputs have an NA. A simultaneous ON at both the FrcOpn and FrcCls inputs is evaluated as a Force Close action.
FrcCls	Force Close	Class: Digital - An ON causes the actuator to drive towards the full closed condition for a time period equal to the full-stroke Drive Time assigned. An OFF or not active (NA) disables the Force Close request.		
HFlowCal	High Flow Calibration	Class: Analog - The value that defines the high flow calibration factor, or the airflow rate equivalent to 1 inch of WC (249 Pascals) of differential pressure (typically supplied by the VAV Pressure Transducer object). Used to automatically calculate the active deadband region where active output drive is not permitted.		A negative value or not active (NA) causes object outputs to hold at their current conditions.
Action	Action	Class: Digital - Determines the actuator rotation (output sequence) required to open and close the VAV terminal box damper by reversing the action of the Drive A and Drive B outputs as follows: • OFF or not active (NA) sets Drive A to close and Drive B to open the actuator. • ON reverses this (sets Drive A to open and Drive B to close the actuator).	OFF or ON	

Table–5.323 VAV Actuator Object Output Properties.

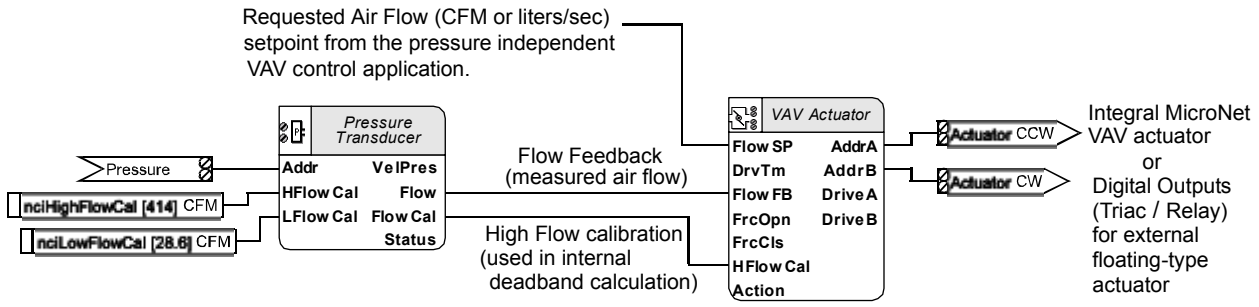
Abbrev.	Name	Class / Description	Valid Values
AddrA	Physical Address A	Class: Analog - Defines the physical hardware address assigned to perform the VAV Actuator Drive A function. A not active (NA) indicates that a physical AddrB is not assigned, and the object updates the Drive A output only.	Dependent on the controller model selected.
AddrB	Physical Address B	Class: Analog - Defines the physical hardware address assigned to perform the VAV Actuator Drive B function. A not active (NA) indicates that a physical AddrB is not assigned, and the object updates the Drive B output only.	Dependent on the controller model selected.

Table–5.323 VAV Actuator Object Output Properties. (Continued)

Abbrev.	Name	Class / Description	Valid Values
DriveA	Drive A	Class: Digital - An ON indicates the actuator is driving towards the Drive A position, otherwise this output is OFF. Refer to the output drive table for the associated actuator direction, based on the assigned Action value.	OFF (0.0) (not active) ON (100.0) (active)
DriveB	Drive B	Class: Digital - An ON indicates the actuator is driving towards the Drive B position, otherwise this output is OFF. Refer to the output drive table for the associated actuator direction, based on the assigned Action value.	OFF (0.0) (not active) ON (100.0) (active)

Applying the Object

The VAV Actuator object is used to control a MicroNet VAV controller actuator, in turn used to position a VAV terminal box damper. The damper actuator regulates air flow in a pressure independent VAV control application. The object controls either the controller's integral actuator (as on an or V2RVx series controller) or for an external floating control type actuator (as used with a MNL-V3RVx series controller). The object is a point-type object, with two hardware addressable outputs for actuator drive close and drive open operation.



Figure–5.173 Typical connection to Pressure Transducer Object.

The VAV Actuator object is typically paired with the VAV Pressure Transducer object as shown in Figure–5.173. The VAV Pressure Transducer object provides air flow feedback (CFM or liters per second) and calibration data. Both the VAV Actuator object and the VAV Pressure Transducer object types are unique to MicroNet VAV controllers. The VAV Actuator object activates the physical and Drive outputs based on differences between the Flow Feedback signal and the requested Flow Setpoint value (received from other VAV application control logic).

Analog Value Inputs

Flow Setpoint: The Flow Setpoint input (FlowSP) connects back to control logic that is properly scaled to match the flow units associated with the feedback input signal (that is, CFM or liters/sec). Typically, the Flow Setpoint originates from a Reset object fed by a Loop object output. A negative or not active (NA) Flow Setpoint value holds object outputs at current positions.

Drive Time: The Drive Time input is used only if the object is for an external actuator, and represents the full-stroke drive time (in seconds) for that actuator. If using the integral MicroNet VAV actuator, the Drive Time input can be left unconnected (not active or NA).

Flow Feedback: The Flow Feedback input receives the measured air flow to compare against the Flow Setpoint, and is typically generated by a VAV Pressure Transducer object. A negative or not active (NA) Flow Feedback value indicates no valid feedback and holds object outputs at current positions.

High Flow Calibration: The High Flow Calibration input is set to a value that represents the air flow rate when input differential pressure is at 1 inch of WC (249 Pascals), and is also typically generated by a VAV Pressure Transducer object. The VAV Actuator object uses this value to internally calculate a deadband region for the drive outputs. This automatic deadband optimizes drive output action (prevents excessive output cycling when near the Flow Setpoint).

Digital Value Inputs

Force Open and Force Close: The Force Open and Force Close inputs allow digital overrides of the normal actuator operation (the present Flow Setpoint value is ignored).

- An ON at the Force Open input causes the VAV actuator to drive towards the full open position for the specified Drive Time. An OFF or not active (NA) disables the Force Open request.
- An ON at the Force Close input causes the VAV actuator to drive towards the full close position for the specified Drive Time. An OFF or not active (NA) disables the Force Close request.
- A simultaneous ON at both the Force Open and Force Close inputs results in the Force Close actuator drive action, with the VAV actuator driving towards the full close position for the specified Drive Time.

Note: Force Open and Force Close inputs remain functional even if other object inputs have a not active (NA).

Action: The Action input defines the rotation scheme for the actuator, allowing reverse operation of the physical and Drive (A and B) outputs.

- An OFF or not active (NA) results in Drive A used as the drive close output, and Drive B used as the drive open output.
- An ON at the Action input results in Drive A used as the drive open output, and Drive B used as the drive close output.

The following tables show how the VAV Actuator object outputs operate with the different Action states OFF or NA Figure-5.324 and ON Figure-5.325.

Table–5.324 Output Operation with Action = OFF or NA.

VAV Actuator Action (Condition)	AddrA / Drive A	AddrB / Drive B
Hold (Feedback = Setpoint)	OFF	OFF
Close (Feedback > Setpoint)	ON	OFF
Open (Feedback < Setpoint)	OFF	ON
Not Allowed	ON	ON

Table–5.325 Output Operation with Action = ON.

VAV Actuator Action (Condition)	AddrA / Drive A	AddrB / Drive B
Hold (Feedback = Setpoint)	OFF	OFF
Close (Feedback > Setpoint)	OFF	ON
Open (Feedback < Setpoint)	ON	OFF
Not Allowed	ON	ON

Note: The Drive A and Drive B outputs reflect the actual physical hardware output (AddrA and AddrB) conditions in all cases except where calculated output travel times are less than the actual object execution time. For this reason, use of the physical hardware outputs is recommended (versus the Drive A and Drive B outputs used with other point type objects such as Binary Output objects).

Normal Operation

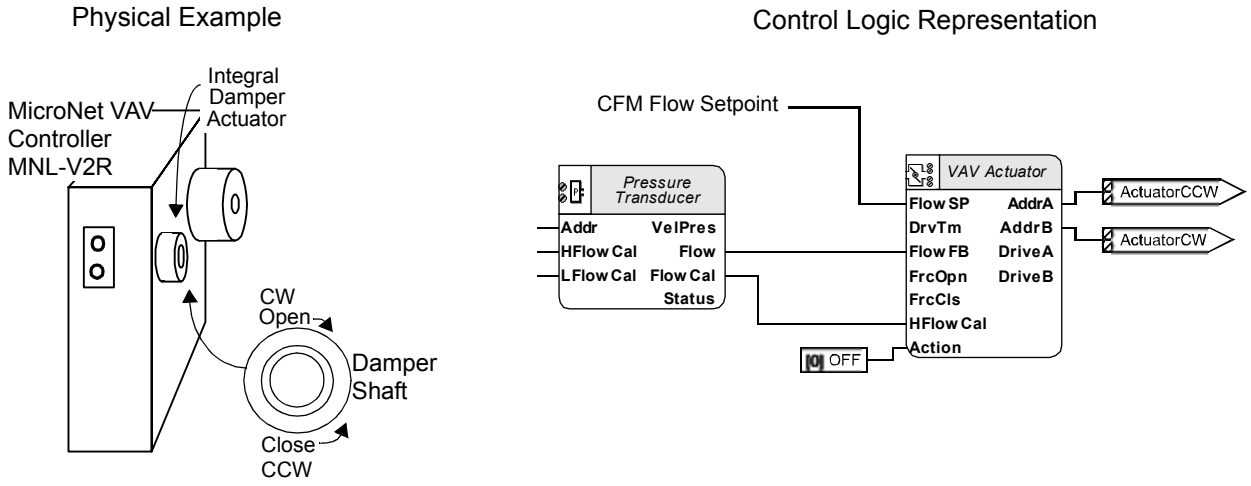
During normal operation, the actual air flow (Flow Feedback) is compared to the requested air flow (Flow Setpoint) to calculate a required actuator drive time and direction. When this difference between actual and requested exceeds the calculated deadband region, the VAV Actuator object drives the outputs in the proper direction to nullify the difference.

When commanded to move to a new air flow setpoint, the VAV Actuator object calculates a required travel time and direction. The actuator continues to drive in this direction until either the Flow Feedback reaches the Flow Setpoint or until the drive time exceeds the user assigned Drive Time (or internally derived Drive Time if the integral controller actuator). Exceeding the Drive Time indicates that the requested air flow cannot be achieved and causes the outputs to deactivate until either the Flow Feedback value changes or the requested Flow Setpoint value changes.

Examples

Integral VAV Damper Actuator

In the following example Figure-5.174 the VAV Actuator object operates the integral VAV damper actuator of an MNL-V2X series VAV controller. The controller is mounted on a VAV terminal box that requires a clockwise (CW) shaft rotation to open the damper, and a counter-clockwise rotation (CCW) rotation to close the damper.

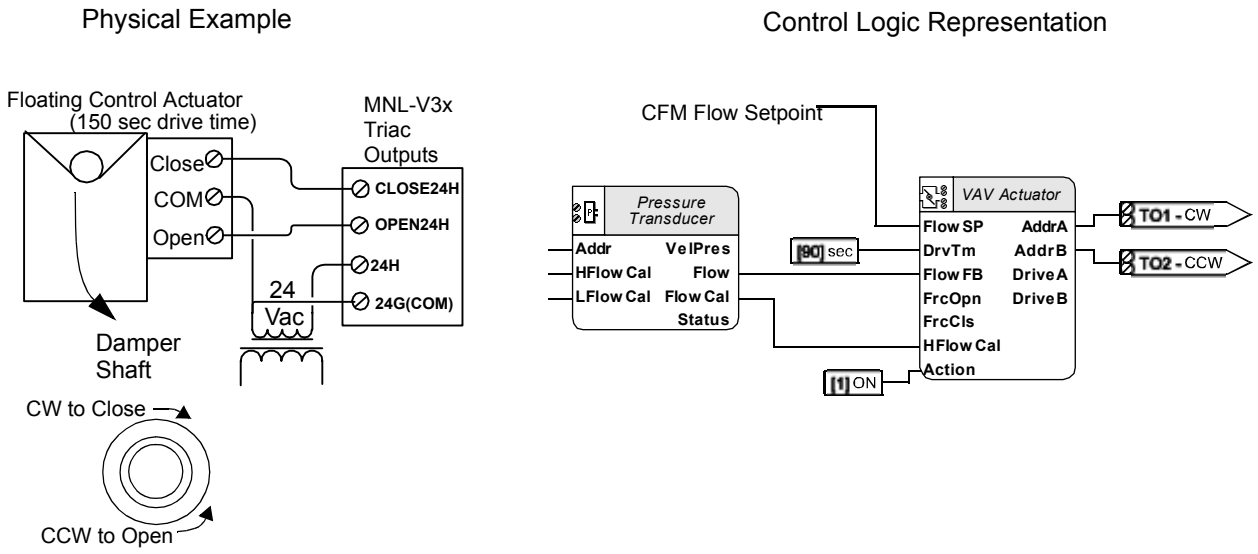


Figure–5.174 VAV Actuator Object for Integral Actuator of MicroNet VAV Controller (MNL-V2R series).

External VAV Damper Actuator

This VAV Actuator object operates an external VAV damper actuator wired to the triac outputs of an MNL-V3X series MicroNet VAV controller. The actuator has a 150 second drive time. The VAV terminal box requires a counter-clockwise (CCW) shaft rotation to open the damper, and a clockwise (CW) rotation to close the damper.

Note: When the VAV Actuator object is used with an external actuator, and the Drive Time is less than 120 seconds, occasionally the digital triac outputs will quickly cycle ON and OFF up to 3 times during the drive cycle. Specifically, this situation can occur whenever a Drive Time of 30 to 90 seconds is used, and when there is a continuous flow error of 3.5 to 4.5 % in the Flow Calibration. To prevent this condition, it is recommended that the Drive Time be set at 120 seconds or greater.



Figure–5.175 VAV Actuator Object for External Actuator Wired to an MNL-V3R VAV Controller.

Chapter 6
LONWORKS Network Data Exchange

This chapter describes LONWORKS Network Variables (NVs) and explains how they allow data to be exchanged between networked I/A Series MicroNet controllers.

The following topics are discussed in this chapter:

LONWORKS Network Architecture

SNVTs

- SNVT Characteristics
- NV Implementation in I/A Series MicroNet Controllers

MicroNet Standard Controllers

- Invensys LONMARK Profiles
- Profile Function
- LONMARK Compliancy
- LONMARK Profile Representations
- WP Tech Representations

MicroNet MN 800 Controllers

- User-Definable Profile
- SNVT Objects
- Mandatory (Default) SNVT Objects
- Adding SNVT Objects

NCI objects

- Applying an NCI Object
- Properties

NVI objects

- Applying an NVI Object
- Properties

NVO objects

- Applying an NVO Object
- Properties

Unit Conversion

Network Management

- Network Address
- Network Variable Binding
- Network Variable Updates
- Other Considerations
- Other NV Data Exchange

Point History Log

Service Pin Operation

Diagnostic Indicators

- MicroNet Controller LEDs
- S-Link Sensor Display

LONWORKS Network Architecture

WP Tech on an I/A Series LonWorks Network

WP Tech communicates with all I/A Series devices, using a LONWORKS FTT-10 Free Topology communications network. LONWORKS is a peer-to-peer communications architecture that allows the direct sharing of data between networked I/A Series MicroNet controllers, and this communication is not affected by the use of routers in the network. A LONWORKS network has a communications speed of 78k baud, using unshielded, twisted-pair cabling, with connections that are not polarity sensitive. Two typical I/A Series LONWORKS networks are illustrated in Figure–6.1.

Data Exchange

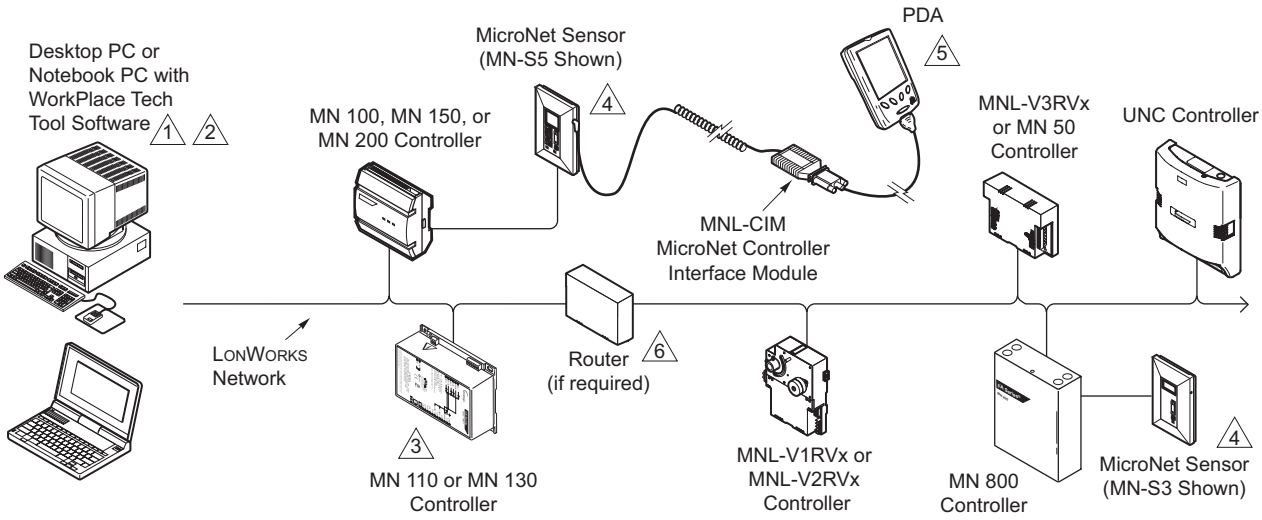
Data exchange is performed through the use of network variables (NVs), which are defined by a set of Standard Network Variable Types (SNVTs). Using network variables, MicroNet controllers can also exchange data directly with other manufacturers' LONWORKS controllers, providing a high level of interoperability. This chapter describes the network variables available in I/A Series MicroNet standard controllers and MN 800 controllers.

Network variables in I/A Series MicroNet controllers fall into three general classes:

- NCIs (Network Configuration Inputs)
- NVIs (Network Variable Inputs)
- NVOs (Network Variable Outputs)

Conceptually, NVs are "data slots," defining the controller's "network image." In any I/A Series MicroNet controller, the collection of these three types of NVs define the controller's "network profile."

In a MicroNet controller, control objects and other resources (sensor tags, schedule tags) can be connected to these data slots to allow sharing of data and control over a LONWORKS network. Data is exchanged between NVIs and NVOs in different devices using peer-to-peer "bindings." Note that the actual engineering of bindings requires a LONWORKS network management tool such as LONMAKER for Windows.



1 A PC can be connected to the LONWORKS TP/FT-10 Network, either directly or through the LONWORKS network jack of a LONWORKS controller or MN-Sxxx Wall Sensor. The PC must have an Echelon® LonTalk® adapter card.

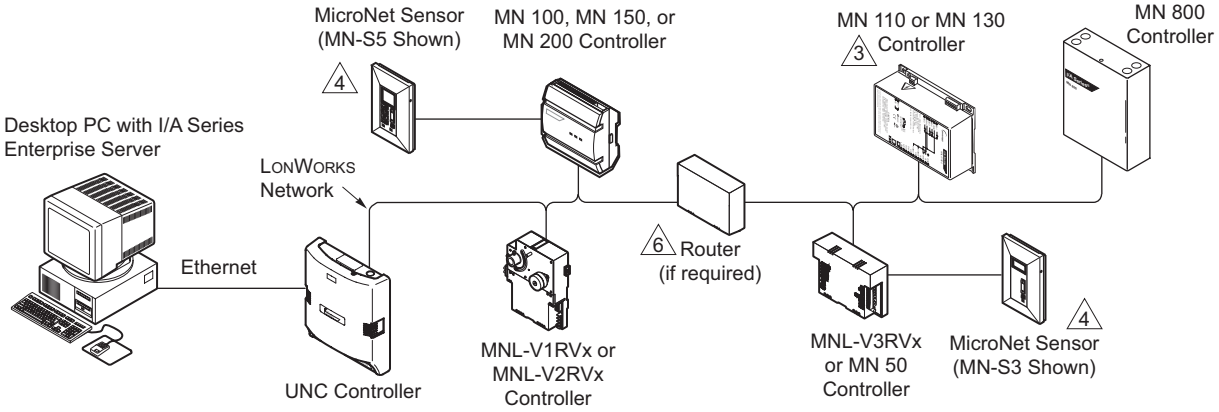
2 Programming any of the I/A Series controllers, or the I/A Series MN 800 controller, requires WorkPlace Tech Tool.

3 This controller is not suitable for exposed mounting on a wall or panel, or in any other easily accessible place due to the possibility of personal contact with the high-voltage terminals. It must be mounted inside a suitable grounded metal enclosure.

4 MicroNet Sensors can be connected to any MN controller.

5 A PDA running the Pocket I/A interface software may be used to communicate with MicroNet I/A Series controllers.

6 When routers are used, WP Tech is able to communicate through them to any of the I/A Series devices on the network.



Figure–6.1 Typical I/A Series LONWORKS Network Architecture.

SNVTs

As previously mentioned, network data exchange is performed through the use of network variables (NVs) that are comprised of a set of LONWORKS Standard Network Variable Types (SNVTs). At the present time, more than 170 SNVTs are defined. Each individually named SNVT provides a predefined value range, resolution, and units. For example, SNVTs have been defined for temperature, pressure, or the digital position of a switch. Using network variables that are based upon SNVTs allows nodes from different vendors to share data because the meaning of the shared variable is understood.

For complete details on all the different types of SNVTs, refer to the LONMARK document, *SNVT Master List and Programmer's Guide*.

Of the 170-plus total SNVTs available, approximately 22 types are supported through network variables within the I/A Series MicroNet standard controllers. In addition, approximately 115 types are supported through the use of SNVT objects when programming an MN 800 controller. A complete list of the SNVTs supported by the I/A Series MicroNet controllers is presented in Appendix B, "Profiles and Network Data (page 615)."

SNVT Characteristics

The following is a brief overview describing the basic SNVT characteristics known as SNVT Type and SNVT Format.

SNVT Types

SNVT types are named using the convention (SNVT_*name*) where the name typically represents the units of measure or the specific function performed. SNVT names are sometimes further delimited, depending on data type, where (SNVT_*name_f*) denotes floating-point data. For each SNVT, the SNVT *type* corresponds to some measurement, range, and resolution, for example:

> SNVT_*temp_p* represents temperature data, where _p denotes a fixed decimal resolution. The range and units of SNVT_*temp_p* are –273.17 to 327.66 °C, with a resolution of 0.01 °C.

Other SNVT types represent a particular function, such as SNVT_*occupancy*, which represents a series of occupancy conditions. Many SNVTs are industry-specific, such as SNVT_*hvac_mode*, which represents a series of mode conditions found within the HVAC industry.

Note: SNVTs use the International System (SI) units (metric) to transfer values across the LON instead of their English equivalents. Any necessary conversion to English units is performed by logic resident in the node or PC interface.

SNVT Formats

SNVTs are represented by three formats: simple SNVTs, structured SNVTs, and enumerated SNVTs.

Simple SNVTs

"Simple SNVT" is a term used to refer to any single-element (single value) SNVT. The *SNVT Master List and Programmer's Guide* defines numerous simple SNVTs, including SNVT_*temp_p*, SNVT_*flow*, SNVT_*count_inc_f*, SNVT_*occupancy*, and others.

Devices exchanging data through a simple SNVT will transfer a single piece of data during active communications of the SNVT value.

Structured SNVTs

A structured SNVT is typically represented by multiple elements (multiple values). Devices exchanging data through a structured SNVT transfer multiple pieces of data during a single communications transfer of the SNVT value, for example:

SNVT_switch has two data fields: State and Value.

- *State* provides a digital indication (either 0 for Off or 1 for On).
- *Value* provides a discrete step indication (0 to 100%). For example, a value of 0%, 33%, 66%, and 100% can respectively represent fan speeds of Off, Low, Medium, and High.

One of the largest structures, SNVT_state, has 16 data fields (Bit-0 through Bit-15), each representing a digital state. Other structured SNVTs have a mixture of other data field types, including simple and/or enumerated types of SNVTs. An NV using a structured SNVT provides an efficient "packaging" of related variables in one message, in which each variable can be separately accessed.

Enumerated SNVTs

An enumerated SNVT is an integer value that represents a set of predefined states or conditions. For example, SNVT_*occupancy* is an enumerated SNVT that defines "occupancy" conditions with a possible value from 0 to 3, as defined below:

0 – OC_OCCUPIED
1 – OC_UNOCCUPIED
2 – OC_BYPASS
3 – OC_STANDBY
255 – OC_NULL

NV Implementation in I/A Series MicroNet Controllers

The following sections describe how network variables (NVs) are implemented in both I/A Series MicroNet standard and MN 800 controllers. The MicroNet standard controllers use *profile tags* that are directly related to the controller and network profile used. The MN 800 controller uses *SNVT objects* that can be programmed to provide a custom network profile.

MicroNet Standard Controllers

Invensys LONMARK Profiles

MicroNet standard controllers (MN 50, 100, 110, 130, 150, 200, VAV series) each contain a pre-defined set of NVs, known as the controller's *profile* (or more specifically, a LONMARK *functional profile*). NVs in the profile are fixed, and define the external data boundaries for the application.

The following LONMARK HVAC functional profiles are available among the MicroNet standard controllers:

- Fan Coil
- Heat Pump
- Roof Top Unit
- VAV (Variable Air Volume)
- Satellite 3 (Roof Top Unit variation)
- Satellite 4 (Roof Top Unit variation)

Models of the MN 50, 100, 150, and 200 controller platforms are available with any of the profiles listed above, except the VAV profile, which is the only profile available for the MicroNet VAV controller model. The MN 110, 130 controller platform is only available with the Fan Coil profile. Each controller's profile is programmed when manufactured. It resides in Read Only Memory (ROM), and cannot be changed or erased.

Profile Function

The profile itself performs no control algorithms—that is the job of the control objects. Instead, a profile can be thought of as a collection of available data slots defined by the items that make up the profile. These items include network variable inputs (NVIs) and network variable outputs (NVOs), plus network configuration properties (NCIs). Together, these profile items suggest a certain application behavior, and provide the "external" or "network image" of the node (controller).

The different types of profile items each serve a different purpose.

- Any data variable input to control logic that is externally sourced (bound) from another node must enter on an NVI of the profile.
- Any data variable output from the control logic to another node (also bound) must exit on an NVO of the profile.
- Any constant value to a configuration property to be accessible by a third-party network management tool must enter on an NCI.

LONMARK Compliancy

Invensys' implementation of each LONMARK profile includes all mandatory and most optional profile items, plus additional "manufacturer-defined" items. Only SNVTs are used for interoperability. Each profile is LONMARK 3.3 compliant.

Note:

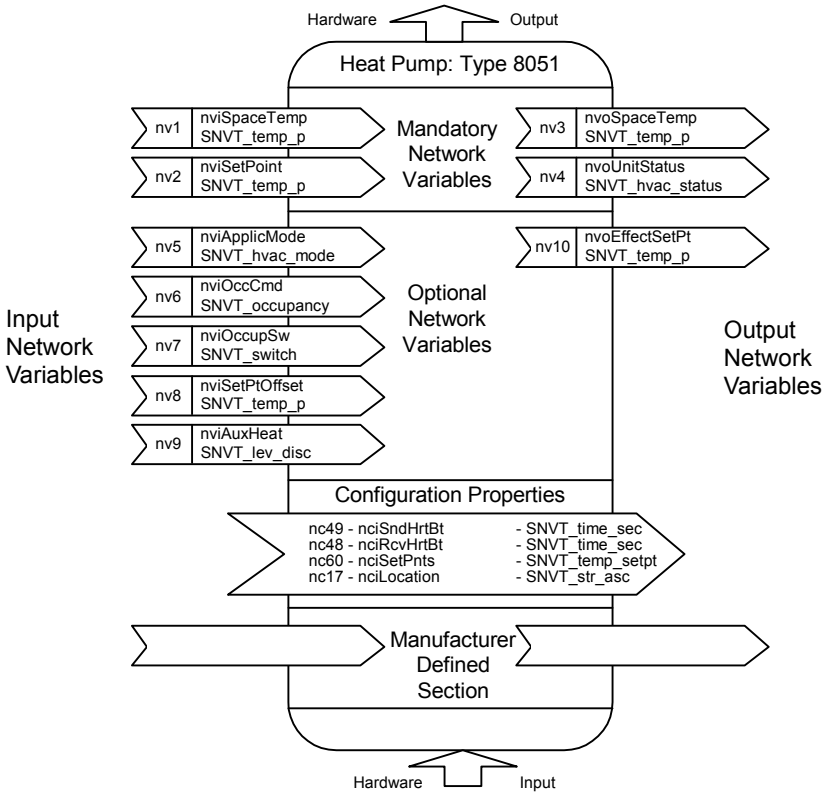
- The Satellite 3 and Satellite 4 profiles are each a variation of the LONMARK 3.3 Rooftop Unit profile, and contain all mandatory items for that profile. However, most items defined as "optional" in the profile are replaced by Invensys-defined items. This allows flexibility when engineering an application that operates differently from any particular profile definition.
- Considerable programming expertise is needed to engineer a custom application that operates in the exact manner documented by the LONMARK functional profile. Use of profile network variables should be done in a manner consistent with the intended use.

LONMARK Profile Representations

A LONMARK profile defines a standardized "network image" that represents a particular device or function. A LONMARK profile is universally represented as a graphical shape, showing all network variable inputs (NVIs) and outputs (NVOs), plus a listing of configuration properties (network configuration inputs, or NCIs).

- NVIs "point in" (enter) as arrows on the left side of the profile.
- NVOs "point out" (exit) as arrows on the right side of the profile.
- Each NVI and NVO is uniquely named using the convention: nvi*Name*, e.g. nviSpaceTemp; and nvo*Name*, e.g. nvoUnitStatus.
- The SNVT type used appears below each nviName and nvoName.
- NCIs are listed together below NVIs and NVOs.

In the example shown in Figure–6.2, the LONMARK Profile 8051 represents the "network image" defined for use within a Heat Pump.

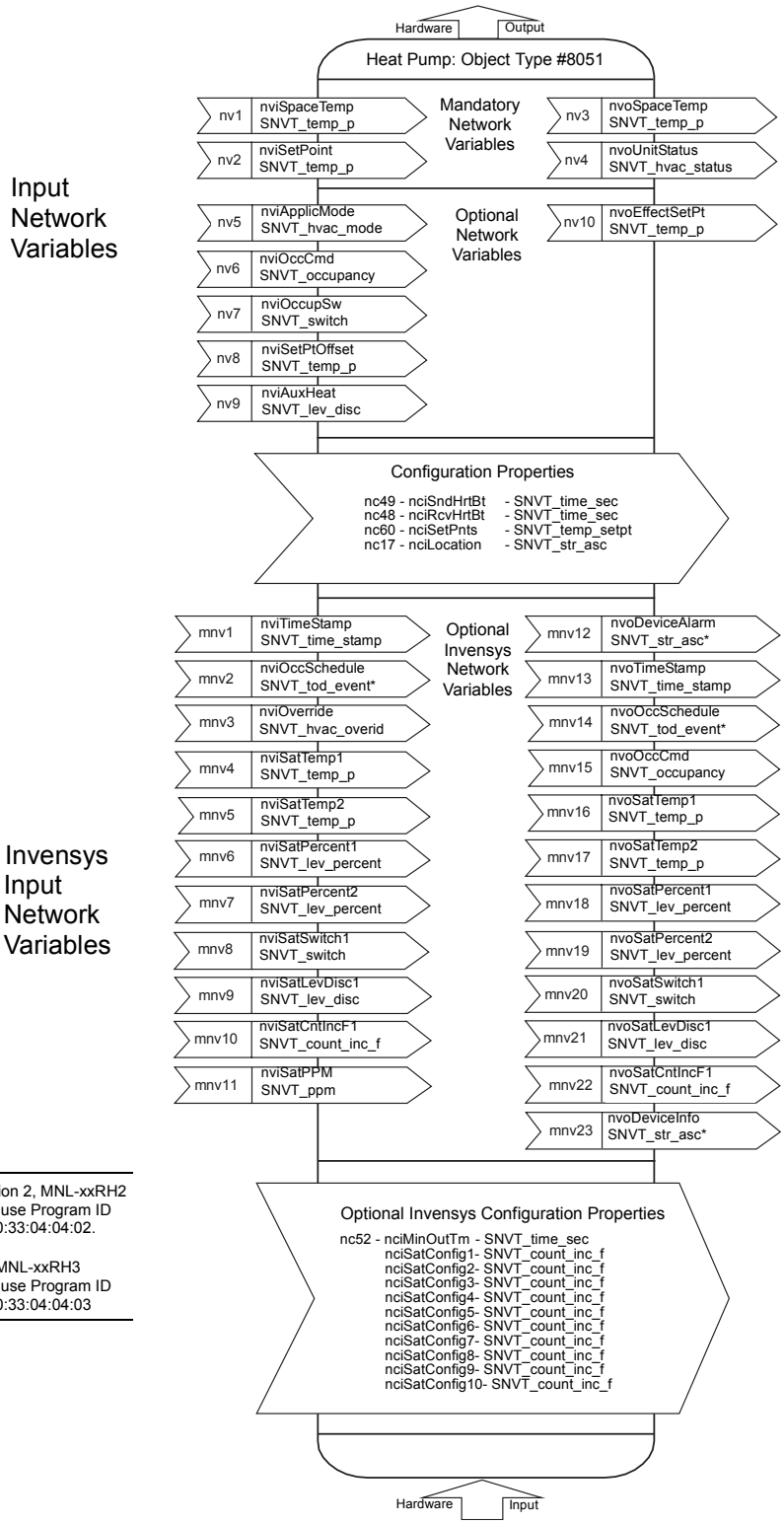


Figure–6.2 Example Graphic Representing a LONMARK Heat Pump Profile.

Note that different sections in the profile correspond to different classes of network variables. These sections are:

- Mandatory Network Variables - at the top of the profile. All of these NVIs and NVOs must be implemented using the defined SNVTs and must operate as described in the LONMARK documentation for that profile.
- Optional Network Variables - below the mandatory network variables. Any number (or all) of these NVIs and NVOs may be implemented. If implemented, they must use the defined SNVTs and operate as described in the LONMARK documentation for that profile.
- Manufacturer Defined - at the bottom of the profile. This section includes additional NVIs and NVOs not included in the standard LONMARK profile, plus additional configuration properties.

  All of the Invensys LONMARK profiles include numerous items in this section, all implemented using SNVTs. These items provide additional flexibility over the mandatory and optional network variables and configuration properties. An example representation of the Invensys LONMARK Heat Pump profile is shown in Figure–6.3.

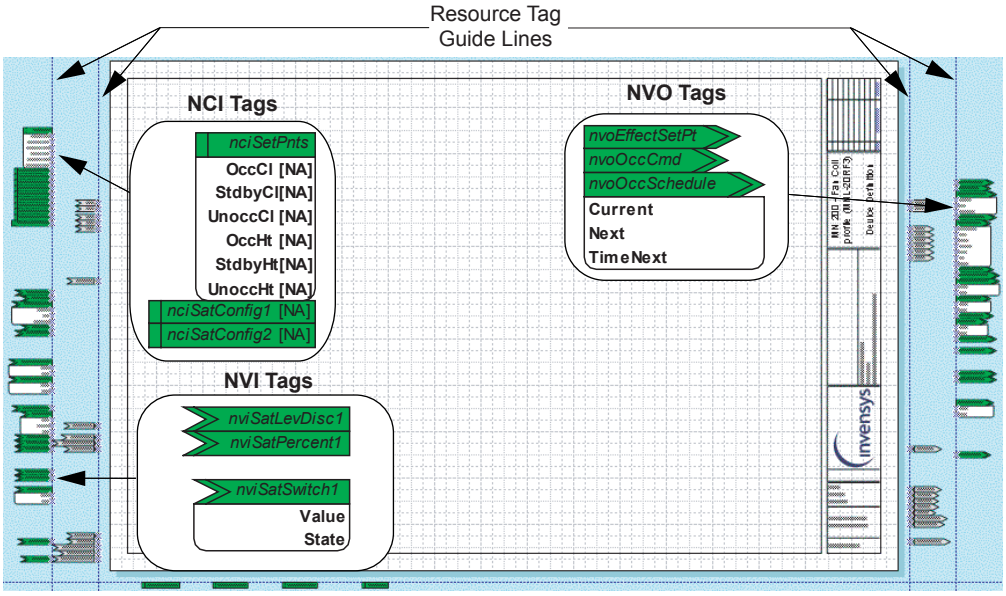


Figure–6.3 Representation of an Invensys LONMARK Heat Pump Profile.

WP Tech Representations

In an application's control logic drawing for a MicroNet standard controller, WP Tech represents the various profile items with individual Profile Tags, using a different shape for each type of profile item:

- NCI Tags - "rectangular-ended" *input* tags, each listing an nci*Name*.
- NVI Tags - "double-arrowed" *input* tags, each listing an nvi*Name*.
- NVO Tags - "double-arrowed" *output* tags, each listing an nvo*Name*.



Figure–6.4 Example Profile Tags for NCIs, NVIs, and NVOs of an Invensys LONMARK Profile.

Profile tags are generated when a particular MicroNet standard controller model is selected in WP Tech. Like other resource tags, profile tags are initially placed on guide lines beyond each end of the control logic page. Like all resource tags, they can be moved into the drawing area and connected to control logic (control objects and tags).

From a control logic perspective, there are two general categories of profile tags:

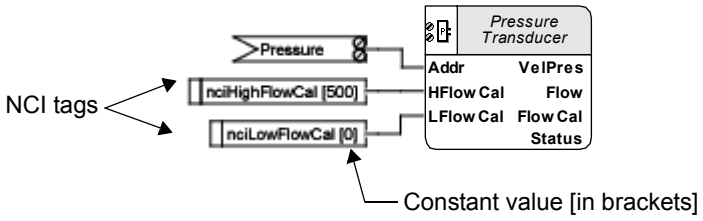
- Input Profile Tags (NCIs and NVIs).
- Output Profile Tags (NVOs).

Input Profile Tags (NCIs and NVIs)

Input profile tags include both the NCI and NVI tags. Each has a single output for connection in the application's control logic. These tags are grouped on the left-side guide line for resource tags.

NCI Tag

An NCI tag is similar to a Constant tag, as it can provide a fixed value to an input of a control object and/or tag. The fixed value is shown inside the tag within brackets beside the nciName.



Figure–6.5 Example NCI Tags in a VAV Application.

Unlike a Constant tag value, an NCI value can also be accessed and/or modified by a third-party network configuration tool or (in the case of the example above) the I/A Series MicroNet VAV Flow Balancing Program. Because of this, WP Tech provides two online functions that are global for all NCIs in an application when "Connected": Write NCI Values and Read NCI Values.

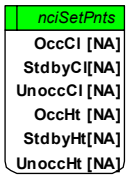
Choosing Read/Write NCI values from the Application menu causes all NCI tags to be written or read as a group. Right clicking an NCI tag allows Reading/Writing the NCI value individually.

A typical application for the Read NCI Values function would be used after running the MicroNet VAV Flow Balancing Program, which calibrates the Pressure Transducer object by writing to selected NCIs Figure-6.5. The Read NCI Value function would update the drawing with the correct NCI values for future downloads (and keep the drawing accurate).

Structured SNVTs: An NCI implemented with a structured SNVT has multiple NCI tags, each with a unique SNVT field (after a period delimiter in the tag's nci*Name*). For example, nciSetPnts is implemented using SNVT_temp_setpt. By LONMARK definition, SNVT_temp_setpt includes these six separate data fields, each corresponding to an individual setpoint:

- occupied_cool
- standby_cool
- unoccupied_cool
- occupied_heat
- standby_heat
- unoccupied_heat

In WP Tech, nciSetPnts is represented by an object with SNVT elements.



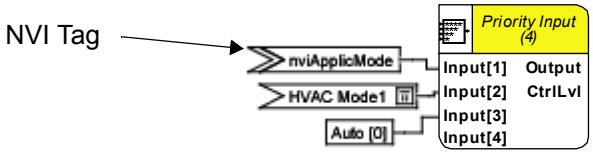
Figure–6.6 Example of NCI Using a Structured SNVT.

SNVT-associated NCI tags, in structured SNVTs, can be individually hidden or shown to make drawings easier to read.

Note: An NCI is typically a constant configuration value that is infrequently changed. Unlike NVI and NVO data, NCI data is stored in the EEPROM of the controller and is retained following a controller power cycle. Because EEPROM supports only a finite number of write operations (approximately 10,000 write cycles), it is important that NCIs are not set up for frequent or continuous updates (changes) when used with third party tools such as I/A Series Niagara.

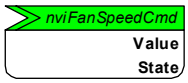
NVI Tag

An NVI tag allows an “externally sourced” data variable to be an input to a control object and/or tag. In other words, it provides a means for receiving a value from another device on the LON.



Figure–6.7 Example NVI Tag Connected to a Control Object.

As with an NCI, an NVI implemented with a structured SNVT has multiple unique SNVT elements. For example, nviFanSpeedCmd is implemented using SNVT_switch. By definition, SNVT_switch includes the two elements State and Value, as shown in Figure–6.8.



Figure–6.8 Example of NVI Using a Structured SNVT.

As with NCI tags, SNVT-associated NVI elements can be individually connected as needed in the control logic application.

Unlike NCI data, which is written to EEPROM, NVI data is written to RAM and may be written frequently as needed. See “Receive Heartbeat Control” on page 545.

Receive Heartbeat Control

nciRcvHrtBt: The NCI tag nciRcvHrtBt (Receive Heartbeat) determines the maximum elapsed time, in seconds, for any NVI value to be considered valid after being updated. An NVI that has not been updated within the timeout specified by the nciRcvHrtBt time value causes the outputs of the associated NVI tag(s) to be set to "not active" (NA) until the next successful update is received. In turn, the “fallback” to "not active" (NA) can be used to provide a means for substituting a default value or condition in the event the network communications to this particular NVI has been lost.

The default nciRcvHrtBt value is NA. This causes objects to not check for updates to the ncis. Invensys recommends that a Receive Heartbeat value be more than twice the send heartbeat value. A typical value is 300 seconds.

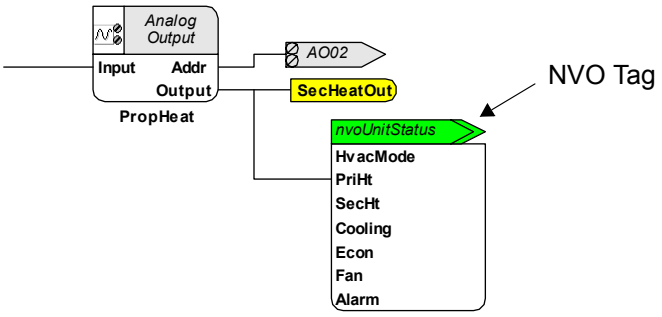
Note: Each NVI can be individually included or excluded in the nciRcvHrtBt function. In WP Tech, each NVI tag shape provides a right-click properties checkbox to access this option. The nciRcvHrtBt setting does not apply to NCIs and has no effect on their operation. The default setting for each NVI is RcvHrtBt enabled.

Output Profile Tags (NVOs)

Output profile tags include NVO tags. Each has a one or more inputs for connection into the application’s control logic. These tags are grouped on the right-side guide line for resource tags.

An NVO allows the data output of a control object or tag to be sent externally as a network variable. In other words, it provides a means for sending controller values to other device(s) on the LON.

Note: Using a network management tool, such as LonMaker or I/A Niagara, the NVO in the profile must be bound to an NVI in another node(s) before actual data is sent. The controllers update the NVO values whenever a Change occurs or when a device polls or requests information from the NVOs.

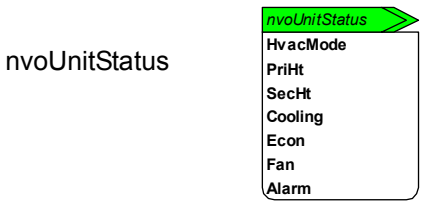


Figure–6.9 Example NVO Connected to a Control Object.

Structured SNVTs: As with an NCI or NVI, an NVO implemented with a structured SNVT has multiple unique SNVT field. For example, nvoUnitStatus is implemented using SNVT_hvac_status. By definition, SNVT_hvac_status includes these seven separate fields:

- mode
- heat_output_primary
- heat_output_secondary
- in_alarm
- cool_output
- econ_output
- fan_output

In the following example Figure-6.10, nvoUnitStatus is represented by an object with seven corresponding SNVT field.



Figure–6.10 Example NVO Tags for an NVO Using a Structured SNVT.

As with NCI and NVI tags, SNVT-associated NVO tags can be individually connected as needed in the control logic application.

Note:

- NVOs not connected into the application are set to the “invalid value” defined by the SNVT type, equivalent to "not active" (NA) in the controller.
- In Rev.3 and later controllers, NVO tags for nvoTimeStamp are not available in the application. Any device that is bound to nvoTimeStamp receives the time/date from the controller’s internal software clock.

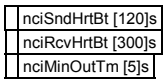
NVO-Related NCIs

Data updates from the control logic in the MicroNet standard controller to connected NVOs are globally governed by settings of two NCIs in the profile:

- nciSndHrtBt (Send Heartbeat)
- nciMinOutTm (Minimum Out Time)

Like the “NVI-related” nciRcvHrtBt (Receive Heartbeat, page 615), NVO-related NCIs are implemented with SNVT_time_sec (seconds). Each has an independent range of from 0 to 6,534 seconds.

These NCIs are represented with separate NCI tags in every MicroNet standard controller application (regardless of profile). They are typically given values but not connected with any other shapes, as shown below.



Figure–6.11 Special NVI and NVO-Related NCIs in a Typical Application

Send Heartbeat Control

nciSndHrtBt: The *Send Heartbeat* setting determines the maximum period of time that is allowed before all bound NVOs are updated, that is, the data values are “re-sent” to all bound nodes. This is a mandatory class NCI for all LONMARK profiles used by I/A Series MicroNet controllers.

The main purpose behind nciSndHrtBt is to ensure that all nodes update values or statuses, regardless of change. This ensures that a device with NVIs, that is setup with a Receive Heartbeat, is guaranteed to receive a valid communication of data within the required timeout period.

The default nciSndHrtBt value is NA, whereby no Send Heartbeat occurs. However, it is usually recommended that a Send Heartbeat value is defined. A typical value may be from 60 to 120 seconds, and should be set to at most one fourth of the Receive Heartbeat time out value.

Minimum Out Time Control

nciMinOutTm: The *minimum out time* setting determines how frequently a rapidly changing value is allowed to force an NVO update. With no minimum out time, an NVO update occurs at every change of value, for every bound NVO. In certain scenarios, this type of unthrottled message traffic on the LON could adversely affect communications between devices.

The default nciMinOutTm value is NA, or no minimum out time. In most cases, a *small* minimum out time is established. A typical value is a few seconds.

Standard Controllers External Interface File (XIF)

An external interface XIF file for a LONWORKS device includes a summary of all the device’s communications parameters, profile items (including network variables), configuration properties, program ID, and implemented SNVTs as well as SNVT order and number. This file may be extracted directly from a LONWORKS device, or it may be acquired from the device’s manufacturer.

I/A Series MicroNet standard controller models are LONMARK-compliant devices, which means that each has a pre-defined XIF file associated with it. Each standard controller stores its XIF file internally, where it may be uploaded and used with a third-party network management tool. Also, because these controllers’ XIF files are pre-defined, they are available for download at the Invensys website, for “offline use” with LONWORKS network management programs like LONMAKER for Windows.

MicroNet MN 800 Controllers

User-Definable Profile

The MN 800 differs from the MicroNet standard controllers in that its network image or profile is completely customizable through the selection and use of “SNVT objects,” which are not available in the other I/A Series MicroNet controllers. The three types of selectable SNVT objects, NCI, NVI, and NVO, offer extended functionality over equivalent profile items in MicroNet standard controllers.

The MN 800 provides the same basic network variable functionality as with the MicroNet standard controllers. However, the MN 800 provides an expanded selection of SNVT types, a more extensive fallback system, and an enhanced set of SNVT object control properties.

Upon generation of a new MN 800 application, WP Tech initializes the drawing with a default set of *mandatory* SNVT objects, which are used to define the network communications characteristics of the controller. These mandatory SNVT objects appear on the controller’s application drawing by default, and cannot be deleted or copied.

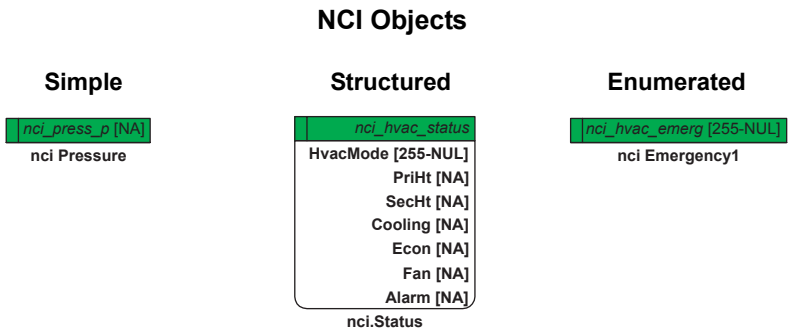
Using the Network Variables stencil, SNVT objects may be selected and added to the application drawing to define the network image of the MN 800 controller. As with the controller objects, SNVT objects are compiled and stored within the controller’s non-volatile EEPROM. SNVT objects may also be uploaded from an existing application in an MN 800.

SNVT Objects

All SNVT objects, whether they are automatically generated (mandatory) or placed on the application drawing by the user, fall into three basic categories: NCI objects, NVI objects, and NVO objects. In all cases, the actual object’s inputs and outputs are based upon the SNVT type used.

NCI Objects (nciType)

An NCI object provides a fixed value (or set of values) to an input of a control object or resource tag, which can be accessed or modified with a third party network configuration tool. The NCI’s output(s) is based upon the SNVT type selected. An NCI based upon a simple (single element) SNVT or an enumerated SNVT will provide one data output. An NCI based upon a structured (multiple element) SNVT will provide a data output for each individual element that is defined. Refer to the “NCI objects” section (page 553) for more details.

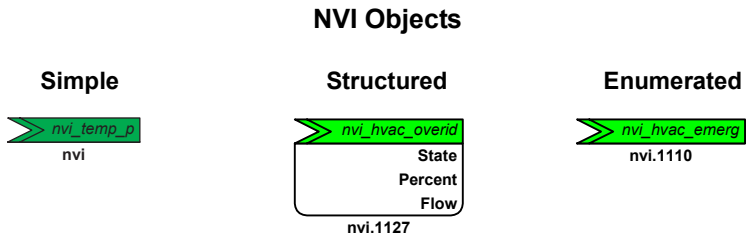


Figure–6.12 Examples of Simple, Structured, and Enumerated NCI Objects.

NVI Objects (nviType)

An NVI object provides a means for receiving a value from another device on the LON, by allowing “externally sourced” network data to be input to a control object or resource tag. The NVI’s output(s) are based upon the SNVT type selected.

An NVI based upon a simple (single element) SNVT or an enumerated SNVT will provide one data output and one default control indication output. An NVI based upon a structured (multiple element) SNVT will provide one data output and one default control indication output for each individual element defined. Additionally, each element includes a default input to allow for a programmed data transition during “fallback” or default conditions. The NVI object also provides a single output to indicate an “out of service” condition. Refer to the “NVI objects” section in this chapter (page 565) for more details.

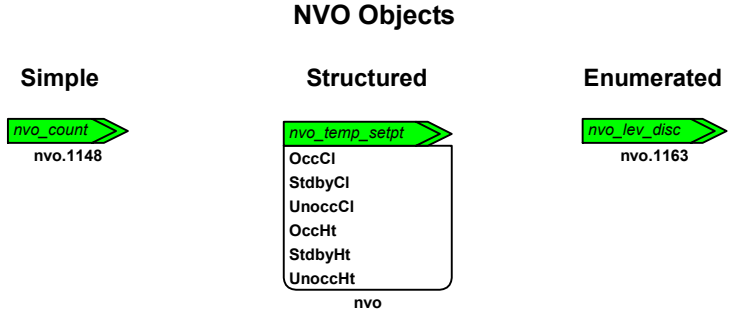


Figure–6.13 Examples of Simple, Structured, and Enumerated NVI Objects.

NVO Objects (nvoType)

An NVO object provides a means for sending a controller’s value to other devices on the LON, by allowing the data output of a control object or resource tag to be sent externally, as a network variable. The NVO’s input(s) are based upon the SNVT type selected.

An NVO based upon a simple (single element) SNVT or an enumerated SNVT will provide one data input and one default input. An NVO based upon a structured (multiple element) SNVT will provide one data input and one default input for each individual element defined. Additionally, the NVO object provides a single “Minimum Propagation” timeout property that can be set to regulate the NVO’s data propagation on the network. Refer to the “NVO objects” section (page 574) for more details.



Figure–6.14 Examples of Simple, Structured, and Enumerated NVO Objects.

Note: It is not necessary to use all the data inputs or outputs of an NCI, NVI, or NVO object that is based on a structured SNVT. Use only those data inputs or outputs that are useful in the application.

Mandatory (Default) SNVT Objects

As previously discussed, the generation of a new MN 800 application causes WP Tech to initialize the drawing with a default set of mandatory SNVT objects. These objects provide the controller with a basic "network image" that is used to setup the controller characteristics (heartbeats, startup delays, timestamp, and location information). These mandatory SNVT objects Table–6.1 appear on the controller's application drawing by default, and cannot be deleted or copied.

Table–6.1 Default MN 800 SNVT Objects.

NCI objects		NVI objects	NVO objects
nciLocation	nciRcvHeartbeatC	nviTimeStamp	nvoDeviceAlarm
nciMinOutTm	nciRcvHeartbeatD		nvoTimeStamp
nciMinPropTm	nciModelNum		
nciNVOStartUpDly	nciSndHeartbeatA		
nciPowerUpDly	nciSndHeartbeatB		
nciRcvHeartbeatA	nciSndHeartbeatC		
nciRcvHeartbeatB	nciSndHeartbeatD		

Refer to the sections, "NCI objects (page 553)," "NVI objects (page 565)," and "NVO objects (page 574)," for detailed descriptions of each mandatory SNVT object.

In addition to the mandatory SNVT objects, WP Tech generates two resource tags, which are linked to the NCI objects that define the controller's start up delays (nciPowerUpDly and nciNVOStartUpDly). This allows the outputs of the individual delays to be connected to the control objects and application as needed.



Figure–6.15 Resource Tags to Define Controller's Start Up Delays.

Adding SNVT Objects

Adding a SNVT object to an application is accomplished in the same manner as adding a control object. To do this, click and drag an NCI, NVI, or NVO object from the Network Variables stencil, to the desired location within the application drawing. A tabbed dialog box appears, to allow for selection of the SNVT type. Once a SNVT type is chosen, WP Tech generates the selected object type, with the appropriate inputs and outputs, at the location where the SNVT object was dropped. In all cases, only the primary inputs or outputs will be visible. The user then right-clicks on the SNVT object and chooses **Customize**, which brings up a dialog box in which the *SNVT Object Name* is displayed. The user modifies this name accordingly, as outlined in the SNVT Object Name section (page 551), which follows.

The user then right-clicks the SNVT object and chooses **Properties**, to bring up the properties editor, which displays the configuration properties for the object. In all cases, SNVT objects are ready "out of the box," which means that WP Tech defaults their configuration properties to a known condition. Additional setup is required only if advanced communications control and indication are needed. Refer to the sections "NCI objects (page 553)," "NVI objects (page 565)," and "NVO objects (page 574)," for a detailed description of available properties, selections, defaults, and usage. Refer to Appendix B for the SNVT types available for use within the MN 800 controller.

Advanced inputs and outputs that provide communications control and indication are hidden unless made visible through use of the object's **Customize** function.

SNVT Object Name and Index Number

NVI, NVO, and NCI objects contain two main properties that are common to the different SNVT object types, SNVT Object Name and NV Index. These configuration properties help define the "network image" within the MN 800 External Interface File (XIF). Mandatory SNVT objects are pre-assigned a unique name and SNVT type. The network variable index numbers are pre-assigned to "-1", allowing WP Tech's compiler to automatically assign index values as needed.

SNVT Object Name

The *SNVT Object Name* represents the "network image" name as seen on the network by third party network installation tools. To enter this name, the user right-clicks on the SNVT object and selects **Customize**, which brings up a dialog box in which the SNVT Object Name is displayed. The SNVT Object Name should be unique and should contain a prefix to help identify the object type: i.e. nvi (NVI object), nvo (NVO object), and nci (NCI object). Names must be 16 characters or less in length, and are limited to alpha, numeric, and underscore characters. For example, an NCI used to represent a low limit could be entered as "nciLow_Limit1." During the compile process, names longer than 16 characters are truncated and any characters other than alpha, numeric, and underscore, such as spaces, periods, or commas, are replaced with underscores.

Note: The use of names longer than 16 characters may result in duplicate names in the external interface (XIF) file and unreliable bindings due to name truncation.

NV Index

The NV Index represents the index position within the controller's "network image" or external interface file. By default, WP Tech is allowed to automatically manage the index numbers. However, NV Indexes can be individually assigned specific index numbers and/or have the index numbers "locked," as required. An index (default –1) indicates that WP Tech may assign an index as needed. Setting a value between 0 and 255 causes WP Tech to lock and hold the index to the value specified. Index values must start at zero and be *contiguous* in the external interface file. WP Tech will attempt to fill any gaps in sequence with SNVT objects that are assigned an index of –1. Gaps that can not be corrected by the WP Tech compiler causes WP Tech to generate an error that must be corrected by the user. WP Tech

has been designed to NOT shift index numbers unless this is absolutely required to fill an index number gap. This means that the addition of a SNVT object should not cause the re-sequencing of index values.

Note: When uploading from an existing MN 800 controller, the SNVTs will have fixed indexes. Deletion of any SNVT will likely force the user to manually re-index the SNVTs. Statistics allow a user to quickly determine the number of SNVTs within a drawing.

MN 800 External Interface File (XIF)

An external interface (XIF) file for a LONWORKS device is a text file that includes a summary of all the device's communications parameters, profile items (including network variables), configuration properties, and implemented SNVTs. This file may be extracted directly from a LONWORKS device, or it may be acquired from the device's manufacturer.

The MN 800 differs from the MicroNet standard controllers, in that its network image or profile is completely customizable through the selection and use of "SNVT objects." When the application is compiled, WP Tech generates and installs the "network image" and generates an external interface (XIF) file, using the drawing name, in the project folder. The XIF file, which may be viewed using Windows Notepad, includes a summary of the node's communications parameters, profile items (including network variables), program ID, configuration properties, and implemented SNVTs.

The WP Tech-generated XIF file is available for "offline use" with LONWORKS network management tools such as LONMAKER for Windows. Also, a downloaded MN 800 controller stores the customized XIF file internally, for "online use."

Caution: Use *extreme caution* when naming a network image or profile for use within the MN 800 controller. Creating two or more controller applications that contain *different* network images, but which have *identical* profile names, *within the same project or installation*, will cause a network management tool such as LONMAKER for Windows to use the *first* uploaded external interface file for *all subsequent installs* of controllers with the same profile name. This causes a mismatch between the network image in the network management tool and those in the controllers. *Unpredictable and improper network operation will occur*.

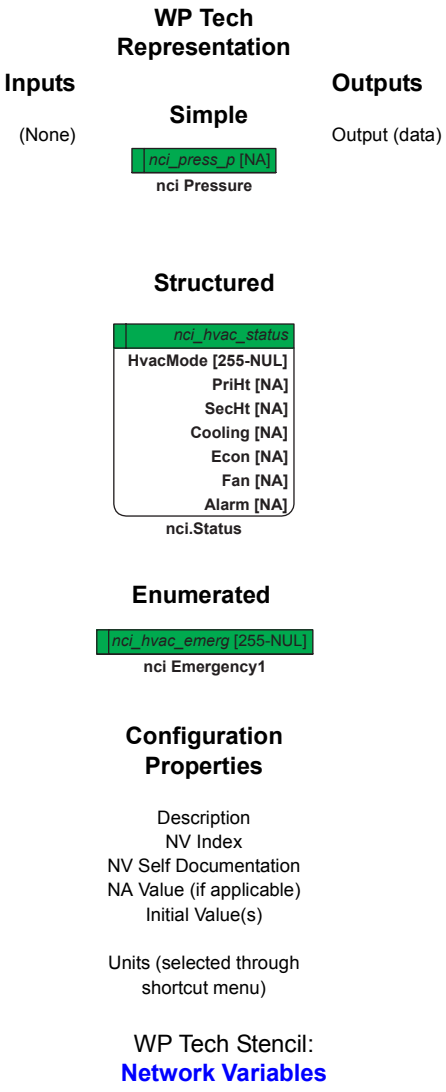
As a Best Practice, each unique MN 800 application should have a unique profile name. The profile name is set in the hardware wizard in the "Use the following program ID" dialog box.

NCI objects

Object Usage: NCI (network configuration input) objects provide LON access to constant values in an MN 800 application. Typically, NCI objects are used whenever access to configuration-type values are required outside of WP Tech from a LONWORKS network management tool. A core set of mandatory (default) NCI objects are included in each MN 800 application.

NCI objects may be copied from the Network Variables stencil to the application drawing, to define the NCI portion of the MN 800 controller's network image.

Device Support:
MN 800



Applying an NCI Object

An NCI object provides a fixed value to an input of a control object or resource tag, that can be accessed or modified by a third party network configuration tool. The NCI object may have one or more outputs, based upon the SNVT type selected. An NCI object based upon a simple (single element) SNVT provides one data output. An NCI based upon a structured (multiple element) SNVT provides one data output for each element that is defined.

An NCI object is applied by first copying the object to the application drawing, then naming the object, and finally by setting the object's properties. These steps are described in the following subsections.

NCI values are stored in EEPROM and are intended to have only 10,000 write operations. Do not link constantly changing values to NCIs.

Adding an NCI Object

Adding an NCI object to the application is accomplished in the same manner as adding a control object. First, click and drag an NCI object from the Network Variables stencil to the desired location on the application drawing. A tabbed dialog box appears to allow for selection of the SNVT type. Once a SNVT type is chosen, WP Tech generates the selected object type, with appropriate outputs, at the location where the NCI object was dropped. See Table–6.5 for a list of the SNVT types that may be used with NCI objects.

Naming an NCI Object

After adding an NCI object to the drawing, the user right-clicks the object to open the shortcut menu and selects **Customize**. This opens a window displaying the Object Name. The default name for an NCI object is “nci.”

Additional instances or duplicates are assigned a numerical suffix to make each object name unique.

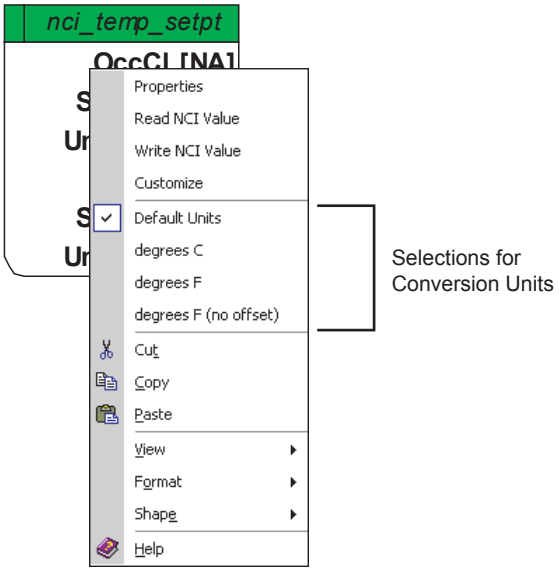
The Object Name should be unique, and should contain a prefix to help identify the object type (i.e. “nci”). The name length must not exceed a total of 16 characters and is limited to alpha, numeric, and underscore characters. For example, a name for an NCI object used to represent a low limit could be entered as “nciLow_Limit1”. Names that exceed 16 characters in length will be truncated to 16 characters during the compile process. Any characters other than alpha, numeric, and underscores (i.e. spaces, periods, commas, etc.) will each be replaced with an underscore during the compile process.

Setting Properties of an NCI Object

Once the NCI object has been named, the user right-clicks the object again to open the shortcut menu, and selects **Properties** to display the Properties Editor. Refer to the “Properties” section (page 556) for a detailed description of available properties, selections, defaults, and usage.

Conversion Selection

By default, the NCI object’s algorithm refers to the engineering units, either metric or English, assigned to the controller. In the hardware wizard, to change conversion units, the user right-clicks the object and chooses a unit of measure from a list located on the shortcut menu, directly below **Customize** Figure-6.16. This list includes all the conversions that are applicable to the NCI object’s selected SNVT type.



Figure–6.16 NCI Object Conversion Units Selection.

If **Default Units** remains selected on the shortcut menu list, the SNVT object's algorithm refers to the engineering units already assigned to the controller in the hardware wizard. Selecting a metric engineering unit, such as **degrees C** in the example in Figure–6.16, causes the algorithm to NOT convert the network value (SNVT types are already based upon System International metric standards). Selecting an English engineering unit, such as **degrees F** in the same example, causes the algorithm to apply a set of predefined conversions for English units, based upon the SNVT type selected for the object. Additional conversions may be predefined, using alternate units, based upon the object's SNVT type. Refer to the "Unit Conversion" section in this chapter (page 585) for more details.

Note: Be sure to select the NCI object's conversion units *before* setting its initial value. Changing the conversion units causes WP Tech to internally use another entry within the database. Therefore, if the units are changed after the initial value is entered, the user must re-enter the initial value.

Properties

Table–6.2 NCI Object Configuration Properties (Typical for All NCI Objects).

Abbrev.	Name	Class / Description	Default	Range/Select
Desc	Description	Class: Character String - Optional user defined descriptor that is available to further describe the object.	—	Up to 250 characters.
Index	NV Index	Class: Analog - Represents the index position of this NV within the controller's "network image" or external interface file (XIF file). By default, WP Tech is allowed to automatically manage the index numbers. However, individual NV indexes can be assigned specific index numbers or have their index numbers "locked", as required. An index of -1 (default) indicates that WP Tech may assign an index as needed. Setting to a value causes WP Tech to lock and hold the index to the integer value specified. ***Note***: "Not active" (NA) defaults to -1 (negative 1).	-1	-1 to 255
SelfDoc	NV Self Documentation	Class: String - The NV Self Documentation contains a maximum of 60 characters of text that may be used to describe the NCI object. The NV Self Documentation is stored as part of the controller's "network image" or external interface file (XIF), for use by third party network management tools. ***Note***: NV Self Documentations, especially those with long text descriptions, must only be used sparingly. Use of this text string directly consumes memory allocated for SNVT objects, thereby decreasing the available memory needed for additional SNVT objects.	—	—
NA	NA Value	Class: Bit - This configuration property determines how the NCI object evaluates a SNVT "maximum value" or the maximum value in the SNVT types range. This selection is only provided on SNVT types that do not have a LONMARK defined "invalid value". A selection of "Use NA" causes the NCI object's algorithm to set the element's output to "not active" (NA) whenever the value on the network reaches the maximum value defined by the SNVT type's range. A selection of "Don't Use NA" causes the NCI object's algorithm to set the output value to the highest maximum controller value allowed (typically 16383) whenever the network value reaches or exceeds the maximum value defined by the SNVT type's range. For example, if an NCI object using SNVT_count_inc as the SNVT type is accessed by a network management tool and changed to a value of 32767, the NCI object's output, based upon this property, will be set as follows: • Use NA – The output is set to a "not active" (NA). • Don't Use NA — The output is set to the maximum controller value allowed (typically 16383). ***Note***: For detailed information on SNVTs, including "invalid" or "not active" (NA) values, see "MN Controllers SNVTs Quick Reference," beginning on page 663 in Appendix B.	Use NA	Use NA Don't Use NA

Table–6.2 NCI Object Configuration Properties (Typical for All NCI Objects).

Abbrev.	Name	Class / Description	Default	Range/Select
Initial Value	Initial Value	Class: Analog / Digital - The NCI object is initialized to this value during the compile and download process. ***Note***: For detailed information on SNVTs, including "invalid" or "not active" (NA) values, see "MN Controllers SNVTs Quick Reference," beginning on page 663 in Appendix B.	—	—

.

Table–6.3 NCI Object Output Properties (Typical for All NCI Objects).

Output Name	Class / Description	Range/Select
Output (data) (Structured Input SNVTs have multiple Outputs; each Output has a unique ID for a specific data field.)	Class: Analog / Digital - The processed network configuration value (or values, if a structured SNVT), each limited to the MicroNet controller number range. Each output represents an available data source for control objects or resource tags in the controller, graphically engineered in WP Tech using a connection wire. Each output produces the defined value in user recognizable units. In the case of NCI objects based on structured SNVTs (those that include more than one element), an Output is provided for each individual element within the structure. ***Note****:* Most of the "default" NCI objects for an MN 800 do not include an output. These objects define global or device-specific parameters.	See Appendix B for details on SNVTs.

Mandatory (Default) NCI Objects

Whenever a new MN 800 application is generated, WP Tech initializes the drawing with a *default set* of mandatory NCI objects that are used to define the characteristics of the controller. These mandatory NCI objects cannot be deleted or copied. Refer to the "MN 800 User-Definable Profile" section in Appendix B of this manual (page 660)" for additional information on the SNVT types used.

Table–6.4 Mandatory NCI Objects in an MN 800 Controller.

NCI Object Type	WP Tech Shape (Default Name Shown)	Typical NCI Application	SNVT Type
nciPowerUpDly (Power Up Delay)		Determines the time, in seconds, that must expire following a controller reset before the MN 800 hardware outputs (Digital Outputs and Analog Outputs) are activated. Controller resets include power cycle, soft resets, and resets caused by the download of an application to the controller. A "not active" (NA) or (nciPowerUpDly) value of less than 8 seconds causes the controller to use an internal power up delay of approximately 8 seconds. The default delay value is 10 seconds.	107
		***Note*:** The digital OFF/ON "result" of this setting is available separately, in a WP Tech resource tag, shown at left. The output of this shape can be connected into control logic as needed. The output is OFF (0) immediately following a reset, until the nciPowerUpDly time has been exceeded. At that point the output changes to ON (100) until the next reset.	
nciNVOStartUpDly (NVO Start Up Delay)		Determines the time, in seconds, that must expire following a controller reset before the first network updates are sent by bound network variable outputs (NVOs). Controller resets include power cycle, soft resets, and resets caused by the download of an application to the controller. A "not active" (NA) or (nciNVOStartUpDly) value of less than 8 seconds causes the controller to use an internal power up delay of approximately 8 seconds. The default value is 10 seconds.	107
		***Note:*** The digital OFF/ON "result" of this setting is available separately, in a WP Tech resource tag, shown at left. The output of this shape can be connected into control logic as needed. The output is OFF (0) immediately following a reset, until the nciNVOStartUpDly time has been exceeded. At that point the output changes to ON (100) until the next reset.	
nciMinPropTm (Minimum Propagation Time)		Defines the minimum time interval, in seconds, that must expire before each NVO object propagates a changed value to the mechanism that actually sends the variables over the network. This parameter can also be described as a propagation filter applied to each NVO object. Each NVO object will use this "global" propagation time value as long as the NVO's Minimum Propagation input has remained unconnected or "not active" (NA). The default (global) Minimum Propagation Time value is "not active" (NA). This value disables the NVOs' Minimum Propagation Time function, allowing the NVOs to propagate a changed value without delay. See the "NVO objects" section in this chapter (page 574) for more details.	107

Table–6.4 Mandatory NCI Objects in an MN 800 Controller.

NCI Object Type	WP Tech Shape (Default Name Shown)	Typical NCI Application	SNVT Type
nciMinOutTm (Minimum Out Time)		Defines the minimum time interval, in seconds, between the sending of updates on the active network, from each propagated NVO. The controller applies this interval to all its bound NVOs. This parameter can also be described as the final filter applied to all NVO updates sent from the controller. The default Minimum Output Time value is 1 second. A "not active" (NA) or a value of zero disables the Minimum Output function, allowing bound NVOs to update without delay.	107
nciRcvHeartbeatA (Receive Heartbeat A)		One of four independent “Receive Heartbeat timers” provided with the MN 800 controller. The Receive Heartbeat, in conjunction with the Send Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). The Receive Heartbeat mechanism is applied to the node’s network variable inputs accordingly. The nciRcvHeartbeatA object defines the maximum time interval, in seconds, that must expire before a non-updated NVI is returned to “out of service” and default control conditions. The value assigned to nciRcvHeartbeatA is used by NVIs that have been set to reference Receive Heartbeat A. The default Receive Heartbeat A value is "not active" (NA). A "not active" (NA) or a value of zero disables the Receive Heartbeat function for any NVIs assigned to use Receive Heartbeat A.	107
nciRcvHeartbeatB (Receive Heartbeat B)		One of four independent “Receive Heartbeat timers” provided with the MN 800 controller. The Receive Heartbeat, in conjunction with the Send Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). The Receive Heartbeat mechanism is applied to the node’s network variable inputs accordingly. The nciRcvHeartbeatB object defines the maximum time interval, in seconds, that must expire before a non-updated NVI is returned to “out of service” and default control conditions. The value assigned to nciRcvHeartbeatB is used by NVIs that have been set to reference Receive Heartbeat B. The default Receive Heartbeat B value is "not active" (NA). A "not active" (NA) or a value of zero disables the Receive Heartbeat function for any NVIs assigned to use Receive Heartbeat B.	107

Table–6.4 Mandatory NCI Objects in an MN 800 Controller.

NCI Object Type	WP Tech Shape (Default Name Shown)	Typical NCI Application	SNVT Type
nciRcvHeartbeatC (Receive Heartbeat C)		One of four independent “Receive Heartbeat timers” provided with the MN 800 controller. The Receive Heartbeat, in conjunction with the Send Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). The Receive Heartbeat mechanism is applied to the node’s network variable inputs accordingly. The nciRcvHeartbeatC object defines the maximum time interval, in seconds, that must expire before a non-updated NVI is returned to “out of service” and default control conditions. The value assigned to nciRcvHeartbeatC is used by NVIs that have been set to reference Receive Heartbeat C. The default Receive Heartbeat C value is "not active" (NA). A "not active" (NA) or a value of zero disables the Receive Heartbeat function for any NVIs assigned to use Receive Heartbeat C.	107
nciRcvHeartbeatD (Receive Heartbeat D)		One of four independent “Receive Heartbeat timers” provided with the MN 800 controller. The Receive Heartbeat, in conjunction with the Send Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). The Receive Heartbeat mechanism is applied to the node’s network variable inputs accordingly. The nciRcvHeartbeatD object defines the maximum time interval, in seconds, that must expire before a non-updated NVI is returned to “out of service” and default control conditions. The value assigned to nciRcvHeartbeatD is used by NVIs that have been set to reference Receive Heartbeat D. The default Receive Heartbeat D value is 300 seconds. A "not active" (NA) or a value of zero disables the Receive Heartbeat function for any NVIs assigned to use Receive Heartbeat D. ***Note:*** By default, all newly created NVIs are set to Receive Heartbeat D.	107

Table–6.4 Mandatory NCI Objects in an MN 800 Controller.

NCI Object Type	WP Tech Shape (Default Name Shown)	Typical NCI Application	SNVT Type
nciSndHeartbeatA (Send Heartbeat A)		One of four independent “Send Heartbeat timers” provided with the MN 800 controller. The Send Heartbeat, in conjunction with the Receive Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). The Send Heartbeat mechanism is applied to the node’s network variable outputs accordingly. The nciSndHeartbeatA object defines the maximum time interval, in seconds, that must expire before a non-changing, bound NVO is sent on the active network. The Send Heartbeat function ensures that a value which has remained unchanged for a period of time is automatically sent on a periodic basis, to update a receiving node or group of nodes. The value assigned to nciSndHeartbeatA is used by NVOs that have been set to reference Send Heartbeat A. The default Send Heartbeat A value is "not active" (NA). A "not active" (NA) or a value of zero disables the Send Heartbeat function for any NVOs assigned to use Send Heartbeat A.	107
nciSndHeartbeatB (Send Heartbeat B)		One of four independent “Send Heartbeat timers” provided with the MN 800 controller. The Send Heartbeat, in conjunction with the Receive Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). The Send Heartbeat mechanism is applied to the node’s network variable outputs accordingly. The nciSndHeartbeatB object defines the maximum time interval, in seconds, that must expire before a non-changing, bound NVO is sent on the active network. The Send Heartbeat function ensures that a value which has remained unchanged for a period of time is automatically sent on a periodic basis, to update a receiving node or group of nodes. The value assigned to nciSndHeartbeatB is used by NVOs that have been set to reference Send Heartbeat B. The default Send Heartbeat B value is "not active" (NA). A "not active" (NA) or a value of zero disables the Send Heartbeat function for any NVOs assigned to use Send Heartbeat B.	107

Table–6.4 Mandatory NCI Objects in an MN 800 Controller.

NCI Object Type	WP Tech Shape (Default Name Shown)	Typical NCI Application	SNVT Type
nciSndHeartbeatC (Send Heartbeat C)		One of four independent “Send Heartbeat timers” provided with the MN 800 controller. The Send Heartbeat, in conjunction with the Receive Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). The Send Heartbeat mechanism is applied to the node’s network variable outputs accordingly. The nciSndHeartbeatC object defines the maximum time interval, in seconds, that must expire before a non-changing, bound NVO is sent on the active network. The Send Heartbeat function ensures that a value which has remained unchanged for a period of time is automatically sent on a periodic basis, to update a receiving node or group of nodes. The value assigned to nciSndHeartbeatC is used by NVOs that have been set to reference Send Heartbeat C. The default Send Heartbeat C value is "not active" (NA). A "not active" (NA) or a value of zero disables the Send Heartbeat function for any NVOs assigned to use Send Heartbeat C.	107
nciSndHeartbeatD (Send Heartbeat D)		One of four independent “Send Heartbeat timers” provided with the MN 800 controller. The Send Heartbeat, in conjunction with the Receive Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). The Send Heartbeat mechanism is applied to the node’s network variable outputs accordingly. The nciSndHeartbeatD object defines the maximum time interval, in seconds, that must expire before a non-changing, bound NVO is sent on the active network. The Send Heartbeat function ensures that a value which has remained unchanged for a period of time is automatically sent on a periodic basis, to update a receiving node or group of nodes. The value assigned to nciSndHeartbeatD is used by NVOs that have been set to reference Send Heartbeat D. The default Send Heartbeat D value is 120 seconds. A "not active" (NA) or a value of zero disables the Send Heartbeat function for any NVOs assigned to use Send Heartbeat D. ***Note:*** By default, all newly created NVOs are set to Send Heartbeat D.	107
nciLocation		Holds up to 30 ASCII text characters and is typically used to describe the physical location of the controller.	36
nciModelNum		Holds up to 30 ASCII text characters and contains the MN 800 model number. The default nciModelNum has been preset as “Invensys MNL-800-101.”	36

NCI Object on Network Variables Stencil

Application

The NCI object can be found on the Network Variables stencil, in an MN 800 application. Adding an NCI object to the application is accomplished in the same manner as adding a control object, by clicking on its icon in the Network Variables stencil and dragging it to the desired location within the application drawing. Once the copied NCI object has been dragged and released, a tabbed dialog box appears to allow for selection of the SNVT type. See Table–6.5 for a list of the SNVTs that may be used with NCI objects. For illustrations of NCI objects' WP Tech shapes, refer to **"**WP Tech Representation (page 553)," at the beginning of this section. For detailed information on SNVTs, including "inactive" values, see "MN Controllers SNVTs Quick Reference," beginning on page 663 in Appendix B.

Simple SNVT: An NCI based upon a simple (single element) SNVT, for example SNVT_time_sec (time in seconds) or SNVT_press_p (pressure), provides one data output and can be used to represent a numeric value.

Enumerated SNVT: An NCI based upon an enumerated SNVT, for example SNVT_hvac_mode or SNVT_occupancy, provides one data output and can be used to represent an event or condition.

Structured SNVT: An NCI based upon a structured (multiple element) SNVT, for example SNVT_switch or SNVT_hvac_overid, provides one data output for each element defined.

Engineering Units

Refer to Table–6.15 for a list of the metric, English, and optional conversion units that are assigned to SNVTs used with the MN 800 controller.

Available SNVTs

Table–6.5 Available LONMARK SNVTs for NCI Objects Copied from Network Variables Stencil.

Category	Available SNVTs		
Simple SNVTs	1 SNVT_amp	35 SNVT_speed_mil	68 SNVT_elec_whr_f
	2 SNVT_amp_mil	36 SNVT_str_asc[a]	71 SNVT_grammage
	3 SNVT_angle	39 SNVT_temp	72 SNVT_grammage_f
	4 SNVT_angle_vel	41 SNVT_vol	75 SNVT_freq_f
	5 SNVT_btu_kilo	42 SNVT_vol_kilo	76 SNVT_freq_hz
	6 SNVT_btu_mega	43 SNVT_vol_mil	77 SNVT_freq_kilohz
	7 SNVT_char_ascii	44 SNVT_volt	78 SNVT_freq_milhz
	8 SNVT_count	45 SNVT_volt_dbmv	79 SNVT_lux
	9 SNVT_count_inc	46 SNVT_volt_kilo	81 SNVT_lev_percent
	13 SNVT_elec_kwh	47 SNVT_volt_mil	82 SNVT_multiplier
	14 SNVT_elec_whr	48 SNVT_amp_f	98 SNVT_pwr_fact
	15 SNVT_flow	49 SNVT_angle_f	99 SNVT_pwr_fact_f
	16 SNVT_flow_mil	50 SNVT_angle_vel_f	100 SNVT_density
	17 SNVT_length	51 SNVT_count_f	101 SNVT_density_f
	18 SNVT_length_kilo	52 SNVT_count_inc_f	102 SNVT_rpm
	19 SNVT_length_micr	53 SNVT_flow_f	104 SNVT_angle_deg
	20 SNVT_length_mil	54 SNVT_length_f	105 SNVT_temp_p
	21 SNVT_lev_cont	55 SNVT_lev_cont_f	107 SNVT_time_sec
	23 SNVT_mass	56 SNVT_mass_f	110 SNVT_area
	24 SNVT_mass_kilo	57 SNVT_power_f	113 SNVT_press_p
	25 SNVT_mass_mega	58 SNVT_ppm_f	123 SNVT_time_min
	26 SNVT_mass_mil	59 SNVT_press_f	124 SNVT_time_hour
	27 SNVT_power	60 SNVT_res_f	125 SNVT_ph
	28 SNVT_power_kilo	61 SNVT_sound_db_f	126 SNVT_ph_f
	29 SNVT_ppm	62 SNVT_speed_f	129 SNVT_smo_obscur
	30 SNVT_press	63 SNVT_temp_f	131 SNVT_temp_ror
	31 SNVT_res	64 SNVT_time_f	138 SNVT_volt_ac
	32 SNVT_res_kilo	65 SNVT_vol_f	139 SNVT_amp_ac
	33 SNVT_sound_db	66 SNVT_volt_f	143 SNVT_turbidity
	34 SNVT_speed	67 SNVT_btu_f	144 SNVT_turbidity_f
Enumerated SNVTs	11 SNVT_date_day	103 SNVT_hvac_emerg	120 SNVT_defr_mode
	22 SNVT_lev_disc	108 SNVT_hvac_mode	121 SNVT_defr_term
	38 SNVT_telcom	109 SNVT_occupancy	122 SNVT_defr_state
	69 SNVT_config_src	118 SNVT_evap_state	145 SNVT_hvac_type
	97 SNVT_override	119 SNVT_therm_mode	
Structured SNVTs	83 SNVT_state	95 SNVT_switch	127 SNVT_chlr_status
	85 SNVT_zerospan	106 SNVT_temp_setpt	128 SNVT_tod_event
	87 SNVT_elapsed_time	111 SNVT_hvac_overid	
	91 SNVT_muldiv	112 SNVT_hvac_status	

a. SNVT type SNVT_str_asc is a special case SNVT available only as an NCI object. An NCI object using SNVT_str_asc provides no output for connection in the control logic. However, such NCIs allow storage of “self documentation” type information of up to 30 characters each.

NVI objects

Object Usage: NVI (network variable input) objects provide inputs for receiving data in an MN 800 application, typically from the network variable outputs (NVOs) of other devices on the LON. One mandatory (default) NVI object (nviTimeStamp) is included in each MN 800 application.

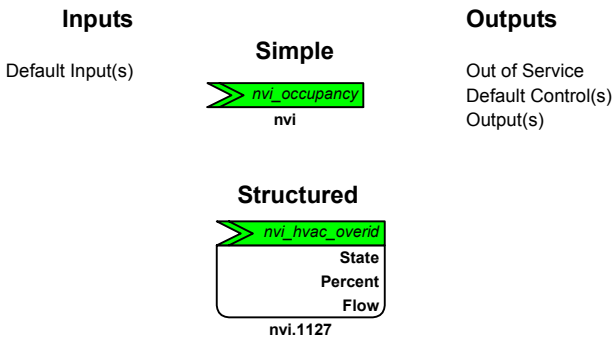
NVI objects may be copied from the Network Variables stencil to the application drawing, to define the NVI portion of the MN 800 controller's network image.

NVI objects have outputs, one or more inputs, and configuration properties. The primary output(s) supply data values received via NV updates. Secondary outputs and input(s) exist for "default-control" reasons, that is, functions associated with receiving "not active" (NA) values or when an "out-of-service" condition is detected. By default, the name of an NVI object appears below its WP Tech shape.

Device Support:

MN 800

WP Tech Representation



Configuration Properties

Description
NV Index
NV Self Documentation
Receive Heartbeat Group Selection
NA Value (if applicable)

Units (selected through shortcut menu)

WP Tech Stencil:
Network Variables

Applying an NVI Object

An NVI object allows externally sourced network data to be input to a control object or resource tag. In other words, it provides a means for receiving a value from another device on the LON. The NVI object may have one or more outputs, based upon the SNVT type selected. An NVI based upon a simple (single element) SNVT provides one data output and one default control indication output. An NVI based upon a structured (multiple element) SNVT provides one data output and one default control indication output for each individual element that is defined. Additionally, each element includes a default input to allow for a programmed data transition during "fallback" or default conditions. The NVI object also provides a single output to indicate an "out of service" condition.

Adding an NVI Object

Adding an NVI object to the application is accomplished in the same manner as adding a control object. First, click and drag an NVI object from the Network Variables stencil to the desired location on the application drawing. A tabbed dialog box appears to allow for selection of the SNVT type. Once a SNVT type is chosen, WP Tech generates the selected object type, with appropriate outputs, at the location where the NVI object was dropped. See Table–6.10 for a list of the SNVT types that may be used with NVI objects.

Note: Only primary level NVI object outputs are immediately visible. Advanced level inputs and outputs such as Default Input, Out of Service, and Default Control, remain hidden. Advanced level inputs and outputs can be made available through the **Customize** selection on the shortcut menu.

Naming an NVI Object

After adding an NVI object to the drawing, the user right-clicks the object to open the shortcut menu and selects **Customize**. This opens a window displaying the Object Name. The default name for an NVI object is “nvi.”

The Object Name must be unique, and should contain a prefix to help identify the object type (i.e. “nvi”). The name length must not exceed a total of 16 characters and is limited to alpha, numeric, and underscore characters. For example, a name for an NVI object used to represent an outdoor air temperature could be entered as “nviOutdoorAir”. Names that exceed 16 characters in length will be truncated to 16 characters during the compile process. Any characters other than alpha, numeric, and underscores (i.e. spaces, periods, commas, etc.) will each be replaced with an underscore during the compile process.

Setting Properties of an NVI Object

Once the NVI object has been named, the user right-clicks the object again to open the shortcut menu, and selects **Properties** to display the Properties Editor. Refer to the “Properties” section (page 567) for a detailed description of available properties, selections, defaults, and usage.

Conversion Selection

By default, the NVI object’s algorithm refers to the engineering units, either metric or English, assigned to the controller through the hardware wizard.. To change conversion units, the user right-clicks the object and chooses a system of units from a list located on the shortcut menu, directly below **Customize.** See Figure–6.16 on page 555. This list includes all the conversions that are applicable to the NVI object’s selected SNVT type.

If **Default Units** remains selected on the shortcut menu list, the SNVT object’s algorithm refers to the engineering units already assigned to the controller through the hardware wizard. Selecting a metric engineering unit, such as **degrees C** in the example in Figure–6.16, causes the algorithm to NOT convert the network value (SNVT types are already based upon System International metric standards). Selecting an English engineering unit, such as **degrees F** in the same example, causes the algorithm to apply a set of predefined conversions for English units, based upon the SNVT type selected for the object. Additional conversions may be predefined, using alternate units, based upon the object’s SNVT type. In all cases, the NVI object algorithm’s output will provide the converted value, which is then used by other objects or resource tags within the application. Refer to the “Unit Conversion” section in this chapter (page 585) for more details.

Properties

Table–6.6 NVI Object Configuration Properties (Typical for All NVI Objects).

Abbrev.	Name	Class / Description	Default	Range/Select
Desc	Description	Class: String - Optional user defined descriptor that is available to further describe the object. The Description is not uploaded with an application.	—	Up to 250 characters.
Index	NV Index	Class: Analog - Represents the index position of this NV within the controller's "network image" or external interface file (XIF file). By default, WP Tech is allowed to automatically manage the index numbers. However, individual NV indexes can be assigned specific index numbers or have their index numbers "locked", as required. An index of -1 (default) indicates that WP Tech may assign an index as needed. Setting to a value other than -1 causes WP Tech to lock and hold the index to the integer value specified. ***Note***: "Not active" (NA) defaults to -1 (negative 1).	-1	-1 to 255
SelfDoc	NV Self Documentation	Class: String - The NV Self Documentation contains a maximum of 60 characters of text that may be used to describe the NVI object. The NV Self Documentation is stored as part of the controller's "network image" or external interface file (XIF), for use by third party network management tools. ***Note***: NV Self Documentations, especially those with long text descriptions, must only be used sparingly. Use of this text string directly consumes memory allocated for SNVT objects, thereby decreasing the available memory needed for additional SNVT objects.	—	—
RcvHrtbtGrp	Receive Heartbeat Group Selection	Class: Analog - The Receive Heartbeat Group defines the source or the mandatory network configuration property used as the NVI's Receive Heartbeat timeout value. Selections include Group A, Group B, Group C, Group D, or None. A selection of "Group A" causes the NVI algorithm to reference nciRcvHeartbeatA as the timeout value. Similarly, "Group B" references nciRcvHeartbeatB, "Group C" references nciRcvHeartbeatC, and "Group D" references nciRcvHeartbeatD. A selection of "None" disables the individual NVI's Receive Heartbeat function. The Receive Heartbeat function determines if network data has been received by the NVI object within the Receive Heartbeat timeout period. An NVI that has not been updated within the timeout period will return the NVI object to default conditions. See NVI object output descriptions for more details.	Group D	None - disables Receive Heartbeat function Group A Group B Group C Group D

Table–6.6 NVI Object Configuration Properties (Typical for All NVI Objects). (Continued)

Abbrev.	Name	Class / Description	Default	Range/Select
NA	NA Value	Class: Bit - This configuration property determines how the NVI object evaluates a SNVT “maximum value” or the maximum value in the SNVT type’s range. This selection is only provided on SNVT types that do not have a LONMARK defined “invalid value”. A selection of “Use NA” causes the NVI object’s algorithm to fall back, passing the element’s Default input value to the output whenever the value on the network reaches the maximum value defined by the SNVT type’s range. A selection of “Don’t Use NA” causes the NVI object’s algorithm to set the output value to the highest maximum controller value allowed (typically 16383) whenever the network value reaches or exceeds the maximum value defined by the SNVT type’s range. For example, if an NVI object using SNVT_count_inc as the SNVT type is accessed by a network management tool and changed to a value of 32767, the NVI object’s output, based upon this property, will be set as follows: • Use NA – The output reflects the value found at the associated Default input. • Don’t Use NA — The output is set to the maximum controller value allowed (typically 16383). ***Note***: For detailed information on SNVTs, including “invalid” or "not active" (NA) values, see “MN Controllers SNVTs Quick Reference,” beginning on page 663 in Appendix B.	Use NA	Use NA Don’t Use NA

Table–6.7 NVI Object Input Properties (Typical for All NVI Objects).

Abbrev.	Name	Class / Description	Range/Select	Notes
Default	Default (Structured Input SNVTs have multiple Default inputs; each Default input has a unique name for a specific data field.)	Class: Analog / Digital - The Default input value is placed at the NVI object’s output during “out of service” or “invalid” value default conditions. Fallback to the default can occur for a variety of reasons, including controller reset, Receive Heartbeat timeout, and receipt of an invalid value on the network. In the case of NVI objects based on structured SNVTs (those that include more than one element), a Default input is provided for each individual element within the structure.	-163.83 to 16383	—

Table–6.8 NVI Object Output Properties (Typical for All NVI Objects).

Abbrev.	Name	Class / Description	Range
OutServ	Out of Service	Class: Digital - Indicates that a loss of communication (Receive Heartbeat timeout) has occurred. A digital ON indicates that an "out of service" condition exists. A digital OFF indicates that communications to this NVI has been established and is being maintained within the Receive Heartbeat timeout (if applicable). A digital OFF on an NVI which has a Receive Heartbeat timeout group selection of “None,” indicates that communications has been established, even if only one message was received. Upon controller reset, the Out of Service output indicates a digital ON until communications to this NVI is established. An NVI object contains only one Out of Service output that applies to all data (single or structured) within the object.	OFF (0.0) ON (100.0)
DefCtrl	Default Control	Class: Digital - Indicates which active input source is providing a value to the Output. A digital ON indicates that the source originates from the Default input. A digital OFF indicates that the source originates from the network. Upon controller reset, the Default Control indicates a digital ON, which means that the Default input is the active input source that is providing a value to the Output. In the case of NVI objects based on structured SNVTs (those that include more than one element), a Default Control output is provided for each individual element within the structure.	OFF (0.0) ON (100.0)
Output	Output (data) (Structured Input SNVTs have multiple Outputs; each Output has a unique name for a specific data field.)	Class: Analog / Digital - Reflects either the processed network input value or the Default input value, based upon current network and default conditions. This output is limited to the MN 800 controller’s number range. A value received through the network that is less than the minimum, or greater than the maximum, controller range values is internally clamped at the minimum or maximum value, respectively. Values are further checked for validity to determine if default conditions are required. Upon controller reset, the Output reflects the value applied to the Default input. In the case of NVI objects based on structured SNVTs (those that include more than one element), an Output is provided for each individual element within the structure.	-163.83 to 16383 NA

Mandatory (Default) NVI Object

When a new MN 800 application is generated, WP Tech initializes the drawing with one *default mandatory* NVI object that is used to synchronize the real time clock within the controller. This mandatory NVI object cannot be deleted or copied. Refer to the "MN 800 User-Definable Profile" section in Appendix B of this manual (page 660)" for more information on the SNVT type used.

Table–6.9 Mandatory NVI objects in an MN 800 controller.

NVI Object Type	WP Tech Shape (Default Name Shown)	Typical NVI Application	SNVT #
nviTimeStamp (Time Stamp Input)		This NVI object provides a means for synchronizing the real time clock within the MN 800 controller. Receipt of valid date and time information at this input will cause the controller to modify the current date and time values used for timekeeping purposes. This NVI object is unique in that there are no inputs or outputs for direct connection to control logic within the MN 800 application. ***Note:*** The MN 800 real time clock is used for time reference by objects that require actual time / day information (i.e. Calendar, Schedule, etc.).	84

NVI Object on Network Variables Stencil

Application

The NVI object is provided on the MN 800 Network Variables stencil. When needed, an NVI object may be added to the application in the same manner as a control object, by clicking on its icon in the Network Variables stencil and dragging it to the desired location within the application drawing. A tabbed dialog box then appears, to allow for selection of the SNVT type, which may be a simple, structured, or enumerated SNVT. Table–6.10 lists the SNVTs that may be used with NVI objects. For illustrations of NVI objects' WP Tech shapes, see "WP Tech Representation (page 565)," at the beginning of this section. For detailed information on SNVTs, including "inactive" values, see "MN Controllers SNVTs Quick Reference," beginning on page 663 in Appendix B.

Simple SNVT: An NVI based upon a simple (single element) SNVT, for example SNVT_temp_p (temperature) or SNVT_press_p (pressure), provides one data output and one default control indication output. Additionally, the single element includes a default input to allow for a programmed data transition during "fallback" or default conditions.

Enumerated SNVT: An NVI based upon an enumerated SNVT, for example SNVT_hvac_mode or SNVT_occupancy, provides one data output representing an event or condition, and one default control indication output. Additionally, the enumerated SNVT includes a default input to allow for a programmed data transition during "fallback" or default conditions.

Structured SNVT: An NVI based upon a structured (multiple element) SNVT, for example SNVT_switch or SNVT_hvac_status, provides one data output and one default control indication output for each individual element defined. Additionally, each element includes a default input to allow for a programmed data transition during "fallback" or default conditions.

Note: In all cases, an NVI object provides a single output to indicate an "out of service" condition. Refer to Table–6.8 for more details.

Advanced Level Inputs and Outputs

As described in the preceding section, primary level NVI object outputs are immediately visible, upon object creation. Advanced level inputs and outputs such as Default Input, Out of Service, and Default Control, are hidden by default but can be made available for use through the **Customize** selection on the shortcut menu.

Default Input

The Default input value is placed at the NVI object's output during "out of service" or "invalid value" default conditions. Fallback to default can occur for a number of reasons, including controller reset, Receive Heartbeat timeout, and receipt of an invalid value on the network. In the case of NVI objects that include more than one element (structured object), a Default input is provided for each individual element within the structure.

The overall operation of the Default input and its fallback operation is directly related to the setting of the NA Value configuration property. This configuration property determines how the NVI object evaluates a SNVT's "maximum value," which is the maximum value in the SNVT type's range. This selection is only provided on SNVT types that do not have a LONMARK defined "invalid value." A selection of "Use NA" causes the NVI object's algorithm to fallback, passing the element's Default input value to the output whenever the value on the network reaches the maximum value defined by the SNVT type's range. A selection of "Don't Use NA" causes the NVI object's algorithm to set the output value to the highest maximum controller value allowed (typically 16383) whenever the network value reaches or exceeds the maximum value defined by the SNVT type's range.

For example, an NVI object that uses SNVT_count_inc as its SNVT type has a range of -32,768 to 32,767. If this NVI object is accessed by a network management tool and set to its maximum value of 32,767, the setting of the NA Value property will determine how the object's output is set, as follows:

- "Use NA" — A network value of 32,767 causes the NVI object's output to reflect the value found on the associated Default input.
- "Don't Use NA" – A network value of 32,767 causes the NVI object's output to be set to the highest maximum controller value allowed (typically 16383).

Note: The NA Value configuration property in all user-defined NVI objects is initialized to the "Use NA" value. This makes it consistent with the I/A Series MicroNet standard controllers, which are configured to always use a predefined invalid value for fallback purposes.

Default Control Output

The Default Control output indicates which active input source is providing a value to the NVI object's Output. A digital ON indicates that the Output value originates from the Default input. A digital OFF indicates that the source originates from within the network.

In the case of NVI objects that include more than one element (structured NVI object), a Default Control output is provided for each individual element within the structure. Upon controller reset, the Default Control indicates a digital ON, which means that the Default input is the active input source that is providing a value to the Output.

Out of Service Output

The Out of Service output indicates that a loss of communications (Receive Heartbeat timeout) has occurred. A digital ON indicates that an “out of service” condition exists. A digital OFF indicates that communication with this NVI has been established and is being maintained within the Receive Heartbeat timeout (if applicable). On an NVI with a Receive Heartbeat timeout group selection of “None,” a digital OFF indicates that communication has been established, even if only one message is received.

Normal Operation

During normal operation, the Default input value is placed at the NVI object’s output whenever fallback occurs. Fallbacks can result from a “Receive Heartbeat timeout” condition, or when a defined invalid network value (based upon the NA Value configuration property) is received.

The receipt of a network variable update by the NVI object causes its algorithm to process the value accordingly and set its Out of Service output to digital OFF. The Receive Heartbeat timer is then started or restarted, based upon the value specified by the referenced Receive Heartbeat Group value (set in a mandatory NCI object).

The Default Control output value is based upon the active source of the output (i.e. fallback or invalid network value conditions). A Default Control of digital OFF indicates that the source originates from within the network. A Default Control of digital ON indicates that the Default input is the source.

If the NVI object fails to receive a network update within the time indicated by the associated Receive Heartbeat timeout, its value is set to the Default input, and the Out of Service and Default Control outputs are set to digital ON.

Reset Operation

Upon controller reset, the NVI object’s Output reflects the value applied to its Default input. Both Out of Service and Default Control outputs will be set to digital ON, and this applies to all resets, including controller power ups and software resets. NVI data that was previously received is not held through a controller reset.

Engineering Units

Refer to Table–6.15 for a list of the metric, English, and optional conversion units that are assigned to SNVTs used with the MN 800 controller.

Available SNVTs

Table–6.10 Available LONMARK SNVTs for NVI Objects Copied from Network Variables Stencil

Category	Available SNVTs		
Simple SNVTs	1 SNVT_amp	35 SNVT_speed_mil	71 SNVT_grammage
	2 SNVT_amp_mil	39 SNVT_temp	72 SNVT_grammage_f
	3 SNVT_angle	41 SNVT_vol	75 SNVT_freq_f
	4 SNVT_angle_vel	42 SNVT_vol_kilo	76 SNVT_freq_hz
	5 SNVT_btu_kilo	43 SNVT_vol_mil	77 SNVT_freq_kilohz
	6 SNVT_btu_mega	44 SNVT_volt	78 SNVT_freq_milhz
	7 SNVT_char_ascii	45 SNVT_volt_dbmv	79 SNVT_lux
	8 SNVT_count	46 SNVT_volt_kilo	81 SNVT_lev_percent
	9 SNVT_count_inc	47 SNVT_volt_mil	82 SNVT_multiplier
	13 SNVT_elec_kwh	48 SNVT_amp_f	98 SNVT_pwr_fact
	14 SNVT_elec_whr	49 SNVT_angle_f	99 SNVT_pwr_fact_f
	15 SNVT_flow	50 SNVT_angle_vel_f	100 SNVT_density
	16 SNVT_flow_mil	51 SNVT_count_f	101 SNVT_density_f
	17 SNVT_length	52 SNVT_count_inc_f	102 SNVT_rpm
	18 SNVT_length_kilo	53 SNVT_flow_f	104 SNVT_angle_deg
	19 SNVT_length_micr	54 SNVT_length_f	105 SNVT_temp_p
	20 SNVT_length_mil	55 SNVT_lev_cont_f	107 SNVT_time_sec
	21 SNVT_lev_cont	56 SNVT_mass_f	110 SNVT_area
	23 SNVT_mass	57 SNVT_power_f	113 SNVT_press_p
	24 SNVT_mass_kilo	58 SNVT_ppm_f	123 SNVT_time_min
	25 SNVT_mass_mega	59 SNVT_press_f	124 SNVT_time_hour
	26 SNVT_mass_mil	60 SNVT_res_f	125 SNVT_ph
	27 SNVT_power	61 SNVT_sound_db_f	126 SNVT_ph_f
	28 SNVT_power_kilo	62 SNVT_speed_f	129 SNVT_smo_obscur
	29 SNVT_ppm	63 SNVT_temp_f	131 SNVT_temp_ror
	30 SNVT_press	64 SNVT_time_f	138 SNVT_volt_ac
	31 SNVT_res	65 SNVT_vol_f	139 SNVT_amp_ac
	32 SNVT_res_kilo	66 SNVT_volt_f	143 SNVT_turbidity
	33 SNVT_sound_db	67 SNVT_btu_f	144 SNVT_turbidity_f
	34 SNVT_speed	68 SNVT_elec_whr_f	
Enumerated SNVTs	11 SNVT_date_day	103 SNVT_hvac_emerg	120 SNVT_defr_mode
	22 SNVT_lev_disc	108 SNVT_hvac_mode	121 SNVT_defr_term
	38 SNVT_telcom	109 SNVT_occupancy	122 SNVT_defr_state
	69 SNVT_config_src	118 SNVT_evap_state	145 SNVT_hvac_type
	97 SNVT_override	119 SNVT_therm_mode	
Structured SNVTs	83 SNVT_state	95 SNVT_switch	127 SNVT_chlr_status
	85 SNVT_zerospan	106 SNVT_temp_setpt	128 SNVT_tod_event
	87 SNVT_elapsed_time	111 SNVT_hvac_overid	
	91 SNVT_muldiv	112 SNVT_hvac_status	

NVO objects

Object Usage: NVO (network variable output) objects provide outputs for sending data from an MN 800 application to network variable inputs (NVIs) of other devices on the LON. Two mandatory NVO objects (nvoTimeStamp and nvoDeviceAlarm) are included in each MN 800 application.

NVO objects may be copied from the Network Variables stencil to the application drawing, to define the NVO portion of an MN 800 controller's network image.

NVO objects have one or more primary inputs (data to be sent), plus secondary inputs and configuration properties. The secondary inputs can be used to individually define the Default value and the Minimum Propagation Time. By default, the name of an NVO object appears below its WP Tech shape.

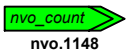
Device Support:
MN 800

WP Tech Representation

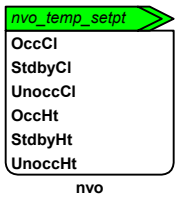
Inputs

Input(s)
Default(s)
Min Propagation Time

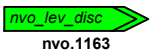
Simple



Structured



Enumerated



Configuration Properties

Description
NV Index
NV Self Documentation
Send Heartbeat Group
Behavior When NA is Present
NA Value

Units (selected through shortcut menu)

Outputs

(None)

WP Tech Stencil:
Network Variables

Applying an NVO Object

An NVO object allows the data output of a control object or resource tag to be sent externally as a network variable. In other words, it provides a means for sending a controller's value to other devices on the LON. The NVO may have one or more inputs, based upon the SNVT type selected. An NVO based upon a simple (single element) SNVT provides two inputs: one Input (data) and one Default. An NVO based upon a structured (multiple element) SNVT will provide these two inputs for each individual element defined. In addition, the NVO object provides a single Minimum Propagation Time input that can be set to regulate the NVO's data propagation on the network.

Adding an NVO Object

Adding an NVO object to the application is accomplished in the same manner as adding a control object. First, click and drag an NVO object from the Network Variables stencil to the desired location on the application drawing. A tabbed dialog box appears to allow for selection of the SNVT type. Once a SNVT type is chosen, WP Tech generates the selected object type, with appropriate inputs, at the location where the NVO object was dropped. See Table–6.14 for a list of the SNVT types that may be used with NVO objects.

Note: Only primary level NVO object inputs are immediately visible. Advanced level inputs such as Default and Minimum Propagation Time remain hidden. Advanced level inputs can be made available through the **Customize** selection on the shortcut menu.

Naming an NVO Object

After adding an NVO object to the drawing, the user right-clicks the object to open the shortcut menu and selects **Customize**. This opens a window displaying the Object Name. The default name for an NVO object is “nvo.”

The Object Name must be unique, and should contain a prefix to help identify the object type (i.e. “nvo”). The name length must not exceed a total of 16 characters and is limited to alpha, numeric, and underscore characters. For example, a name for an NVO object used to represent an outdoor air temperature could be entered as “nvoOutdoorAir”. Names that exceed 16 characters in length will be truncated to 16 characters during the compile process. Any characters other than alpha, numeric, and underscores (i.e. spaces, periods, commas, etc.) will each be replaced with an underscore during the compile process.

Setting Properties of an NVO Object

Once the NVO object has been named, the user right-clicks the object again and selects **Properties**, to display the Properties Editor. Refer to the “Properties” section (page 576) for a detailed description of available properties, selections, defaults, and usage.

Conversion Selection

By default, the NVO object’s algorithm uses the engineering units, either metric or English, assigned to the controller by the Hardware Wizard. To change engineering units, the user right-clicks the object and chooses from a list of units located on the shortcut menu, directly below **Customize.** See Figure–6.16 on page 555. This list includes all the conversions that are applicable to the NVO object’s selected SNVT type.

If **Default Units** remains selected on the shortcut menu list, the SNVT object’s algorithm refers to the engineering units already assigned to the controller. Selecting a metric engineering unit, such as **degrees C** in the example in Figure–6.16, causes the algorithm to NOT convert the network value (SNVT types are already based upon System International metric standards). Selecting an English engineering unit, such as **degrees F** in the same example, causes the algorithm to apply a set of predefined conversions for English units, based upon the SNVT type selected for the object. Additional conversions may be predefined, using alternate units, based upon the object’s SNVT type. In all cases, the NVO object’s algorithm will convert the value received from the objects or resource tags within the application, to the appropriate units as defined by the SNVT type. Refer to the “Unit Conversion” section in this chapter (page 585) for more details.

Properties

Table–6.11 NVO Object Configuration Properties (Typical for All NVO Objects).

Abbrev.	Name	Class / Description	Default	Range/Select
Desc	Description	Class: String - Optional user defined descriptor that is available to further describe the object. The Description is not uploaded with an application.	—	Up to 250 characters.
Index	NV Index	Class: Analog - Represents the index position of this NV within the controller's "network image" or external interface file (XIF file). By default, WP Tech is allowed to automatically manage the index numbers. However, individual NVs can be assigned specific index numbers or have their index numbers "locked", as required. An index of -1 (default) indicates that WP Tech may assign an index as needed. Setting to a value causes WP Tech to lock and hold the index to the integer value specified. ***Note***: "Not active" (NA) defaults to -1 (negative 1).	-1	-1 to 255
SelfDoc	NV Self Documentation	Class: String - The NV Self Documentation contains a maximum of 60 characters of text that may be used to describe the NVO object. The NV Self Documentation is stored as part of the controller's "network image" or external interface file (XIF), for use by third party network management tools. ***Note***: NV Self Documentations, especially those with long text descriptions, must only be used sparingly. Use of this text string directly consumes memory allocated for SNVT objects, thereby decreasing the memory available for additional SNVT objects.	—	—
SndHrtbtGrp	Send Heartbeat Group	Class: Analog - The Send Heartbeat Group defines the source of the mandatory network configuration property used as the NVO's Send Heartbeat timeout value. Selections include Group A, Group B, Group C, Group D, or None. A selection of "Group A" causes the NVO's algorithm to reference nciSndHeartbeatA as the timeout value. Similarly, "Group B" references nciSndHeartbeatB, "Group C" references nciSndHeartbeatC, and "Group D" references nciSndHeartbeatD. A selection of "None" disables the individual NVO's Send Heartbeat function. The Send Heartbeat function ensures that a value, which has remained unchanged for a period of time, is automatically sent to the appropriate receiving node or group of nodes on the network. Working in conjunction with the targeted node's Receive Heartbeat, the Send Heartbeat mechanism ensures that communication is established and regularly maintained, to prevent the receiving node from returning to fallback conditions. The Send Heartbeat function is actively applied only to bound network variable outputs.	Group D	None - disables Send Heartbeat function Group A Group B Group C Group D

Table–6.11 NVO Object Configuration Properties (Typical for All NVO Objects). (Continued)

Abbrev.	Name	Class / Description	Default	Range/Select
NA Behavior	Behavior When NA is Present	Class: Bit - Determines how the NVO object handles the Send Heartbeat function and default input values when "not active" (NA) is applied to the object's primary Input. During primary input "not active" (NA) conditions, the NVO object algorithm uses the NA Behavior configuration property to determine whether to continue or interrupt the sending of the default value to other devices on the network. In the case of structured SNVTs, all elements within the structure must have primary inputs of "not active" (NA) for the NA Behavior to be enforced. When set to "Continue Heartbeat," NA Behavior causes the NVO object's algorithm to continue sending the default value (Default or NA Value) resulting from a change of the Default input value or a Send Heartbeat timeout. A change of value at the Default input sends the Default input value to the network with all Minimum Propagation timeouts applied. A default value that has remained unchanged for a period of time will continue to be sent on a regular basis, as defined by the associated Send Heartbeat timeout. When set to "Interrupt Heartbeat," NA Behavior causes the NVO object's algorithm to suspend the sending of the Default value, regardless of a default value change or a Send Heartbeat timeout. ***Note:*** In all cases, NVO values are kept up to date with current input or default values, for polling purposes.	Continue Heartbeat	Continue Heartbeat Interrupt Heartbeat
Value.NA or (if multi-element) *ElementName*.NA	NA Value	Class: Analog - Represents the network value that is used whenever both the Input and Default input are found to be "not active" (NA). When an NVO object is created, its NA Value is initialized to an NA that represents a predefined value in accordance with the SNVT type used. Active propagation of the default value to the network is dependent upon the setting of the NA Behavior configuration property. The NA Value can be changed to any value within the SNVT type's range. However, it is recommended that the predefined NA value be used. In the case of structured SNVTs, each element within the structure is assigned an NA Value. The I/A Series MicroNet standard controller NVIs can be configured to use these predefined invalid values (from an MN 800), for fallback purposes. ***Note:*** In all cases, NVO values are kept up to date with current input or default values, for polling purposes.	Default is based upon SNVT type used. See "MN Controllers SNVTs Quick Reference" section in Appendix B of this manual (page 663), for detailed information on SNVTs, including "invalid" or "not active" (NA) values.	

Table–6.12 NVO Object Input Properties (Typical for All NVO Objects).

Abbrev.	Name	Class / Description	Range/Select	Notes
Input	Input (Structured Output SNVTs have multiple primary Inputs; each primary Input has a unique name for a specific data field.)	Class: Analog / Digital – The primary input for the NVO object. This Input provides a network variable output interface to controllers' object outputs or resource tags. The algorithm monitors the Input for a valid value and generates an active value for network propagation and use. The NVO object converts the application data into the proper network data format, in accordance with the SNVT type selected, and sends the value as required. In the case of structured NVO objects (those with more than one element), an individual primary Input is provided for each element within the structure.	-163.83 to 16,383	A value of "not active" (NA) causes the NVO to fallback to default conditions (the Default input or the NA Value configuration property).
Default	Default (Structured Output SNVTs have multiple Default inputs; each Default input has a unique name for a specific data field.)	Class: Analog / Digital – Used by the network variable output algorithm whenever the Input (primary input) value is found to be "not active" (NA). The algorithm monitors the Default input for a valid value and uses this value to generate an active value for network propagation and use. In the case of structured NVO objects (objects that include more than one element), a Default input is provided for each individual element within the structure.	-163.83 to 16,383	A Default input of "not active" (NA) causes the algorithm to use the NA Value configuration property as the NVO's active value.
MinPropTm	Minimum Propagation Time	Class: Analog – Defines the time interval (in seconds) that must expire before the NVO object is allowed to propagate a changed value to the task that handles the actual sending of variables on the network. This property can also be described as a filter that can be used to limit the amount of network traffic generated by a bound NVO. A Minimum Propagation Time input of "not active" (NA) causes the NVO object's algorithm to reference the predefined "global" Minimum Propagation Time value (nciMinPropTm). An NVO object contains only one "Minimum Propagation Time" input that applies to all data (single or structured) within the object.	0 to 10,000 seconds	A local or global value of 0 (zero) disables the Minimum Propagation Time function, allowing the NVO object to propagate a changed value without delay.

Mandatory (Default) NVO Objects

Upon generation of a new MN 800 application, WP Tech initializes the drawing with a *default* set of *mandatory* NVO objects that are used to output controller timekeeping and alarm information. These NVO objects cannot be deleted or copied. Refer to the "MN 800 User-Definable Profile" section in Appendix B of this manual (page 660) for additional information on the SNVT types used.

Table–6.13 Mandatory NVO Objects in an MN 800 controller.

NVO Object Type	WP Tech Shape (Default Name Shown)	Typical NVO Application	SNVT #
nvoTimeStamp (Time Stamp Output)		Allows the date and time generated by the real time clock within the MN 800 controller to be shared with other LON devices on the network. The nvoTimeStamp NVO object is represented by SNVT type SNVT_time_stamp. Binding nvoTimeStamp to another device causes the MN 800 to send the time stamp value at a rate defined by the Minimum Propagation Time property. This property's time value defines the time interval (in seconds) that must expire before the timestamp is allowed to propagate. The default timeout value is 60 seconds. Setting the local Minimum Propagation Time to a value that is less than 1 second causes a bound nvoTimeStamp to propagate at 1 second intervals. Setting the local Minimum Propagation Time value to NA causes the object's algorithm to use the value defined by the global nciMinPropTm. For this case, the same rules apply whereby a global Minimum Propagation Time value that is less than 1 second, or NA, causes a bound nvoTimeStamp to propagate at 1 second intervals. ***Note:*** A bound nvoTimeStamp will not be propagated if the controller's time is invalid or unsynchronized.	84
nvoDeviceAlarm (using SNVT_str_asc)		Provides an ASCII character string that contains service alarm and user alarm information. ***Note:*** This object is reserved for future use. It is not currently implemented.	36

NVO Object on Network Variables Stencil

Application

The NVO object is provided on the Network Variables stencil in an MN 800 application. When needed, an NVO object may be added to the application in the same manner as a control object, by clicking on its icon in the Network Variables stencil and dragging it to the desired location within the application drawing. A tabbed dialog box then appears, to allow for selection of the SNVT type, which may be a simple, structured, or enumerated SNVT. Table–6.14 lists the SNVTs that may be used with NVO objects. For illustrations of NVO objects' WP Tech shapes, see "WP Tech Representation," on page 574. For detailed information on SNVTs, including "inactive" values, see "MN Controllers SNVTs Quick Reference," beginning on page 663 in Appendix B.

Simple SNVT: An NVO based upon a simple (single element) SNVT, for example SNVT_temp_p (temperature) or SNVT_press_p (pressure), provides one data input and one default input.

Enumerated SNVT: An NVO based upon an enumerated SNVT, for example SNVT_hvac_mode or SNVT_occupancy, provides one data input and one default input.

Structured SNVT: An NVO based upon a structured (multiple element) SNVT, for example SNVT_switch or SNVT_hvac_status, provides one data input and one default input for each individual element that is defined.

Note: In all cases, the NVO object provides a single Minimum Propagation Time (timeout) input that can be set to regulate the NVO object's data propagation on the network. The Minimum Propagation Time property can also be described as a filter that can be used to limit the amount of network traffic generated by an individual NVO. Refer to Table–6.12 for details.

Advanced Level Inputs

As described in the preceding section, primary level NVO object inputs are immediately visible upon object creation. Advanced level inputs such as Default and Minimum Propagation Time are hidden by default but can be made available for use through the **Customize** selection on the shortcut menu.

Default

The Default input value is used by the NVO object's algorithm whenever the primary input is found to be "not active" (NA). The algorithm monitors the Default input for a valid value and uses this value to generate an active value for network propagation and use.

The NVO object converts the active default value into the proper network data format for the network variable type selected. The NVO object propagates this value according to the setting of the NA Behavior configuration property. An NA Behavior setting of "Continue Heartbeat" causes the algorithm to continue propagating the default value (Default or NA Value) resulting from a change of the Default input value or a Send Heartbeat timeout. A NA Behavior setting of "Interrupt Heartbeat" causes the algorithm to suspend the sending of the Default value, regardless of a Default input value change or a Send Heartbeat timeout. In the case of objects that include more than one element (structured object), a Default input is provided for each individual element within the structure.

A Default input of “not active” (NA) causes the NVO object’s algorithm to use the NA Value configuration property as the active value.

The Default value for the NA Value configuration property varies by SNVT type. When an NVO object is created, its NA Value is initialized to a set of predefined values that are based upon the SNVT type used and engineering unit selected. The I/A Series MicroNet controllers’ NVIs can be configured to use these predefined invalid values for fallback purposes.

For SNVT types that have a LONMARK defined invalid value, NA Value is initialized to this value. For SNVT types that do not have a LONMARK defined invalid value, NA Value is initialized to a set of predefined values, typically the highest value in the SNVT type’s range. The NA Value can be changed to any value within the SNVT type’s range. However, it is recommended that the predefined NA Value be used. In the case of structured SNVTs, each element within the structure is assigned an NA Value. For detailed information on SNVTs, including “inactive” values, see “MN Controllers SNVTs Quick Reference,” beginning on PAGE 663 in Appendix B.

Minimum Propagation Time

The Minimum Propagation Time input defines the time interval (in seconds) that must expire before the NVO object is allowed to propagate a changed value to the task that handles the actual sending of variables on the network. This property can also be described as a filter that can be used to limit the amount of network traffic generated by a bound NVO.

A Minimum Propagation Time input of “not active” (NA) causes the NVO object’s algorithm to reference the predefined “global” Minimum Propagation Time value (nciMinPropTm). The Minimum Propagation Time value can range between 0 and 10,000 seconds. A local or global value of 0 (zero) disables the Minimum Propagation Time function, allowing the NVO object to propagate a changed value without delay.

An NVO object contains only one “Minimum Propagation Time” input that applies to all data (single or structured) within the object.

Normal Operation

The NVO object’s algorithm monitors the primary input(s) for valid values and updates the active value accordingly. The active propagation of the value to the network is based upon Minimum Propagation Time and Send Heartbeat timeout conditions. For actively changing Input values, the Minimum Propagation Time must expire before another active value can be propagated to the network. An active Input value that has remained unchanged for a period of time is automatically propagated at a time specified by the Send Heartbeat timeout.

Detection of an Input value of “not active” (NA) causes the NVO object to fallback and update the active value to reflect default conditions (Default input or NA Value). When a “not active” (NA) condition exists at the primary input, the algorithm uses the NA Behavior configuration property to determine whether to continue propagation of the default value (Default input or NA Value) during default fallback conditions. If the NVO object is based on a structured SNVT, all elements within the structure must have primary inputs of not active (NA).

An NA Behavior setting of “Continue Heartbeat” causes the NVO object’s algorithm to continue propagating the default value due to a change of value or a Send Heartbeat timeout. A change of value at the Default input will be propagated to the network with Minimum Propagation timeouts applied. A default value that has remained unchanged for a period of time will continue to be propagated on a regular basis, as defined by the Send Heartbeat time property.

An NA Behavior setting of “Interrupt Heartbeat” causes the algorithm to suspend network propagation of the default value regardless of a change of value or a Send Heartbeat timeout.

In all cases, the NVO object maintains the latest value for polling and network propagation purposes. Also, the Minimum Propagation and Send Heartbeat functions are only applied to NVOs that are bound.

As a rule, the user should always set the assigned Send Heartbeat to a time value that is greater than the assigned Minimum Propagation Time value. An assigned Send Heartbeat timeout of zero disables the Send Heartbeat function, causing the NVO object’s algorithm to use the Minimum Propagation Time value as needed.

An assigned Send Heartbeat timeout that is greater than zero enables the Send Heartbeat function, causing the NVO object’s algorithm to compare the Minimum Propagation Time value to the Send Heartbeat value. A Minimum Propagation Time found to exceed the assigned Send Heartbeat time causes the object to internally limit the Minimum Propagation Time to a value that is equal to the assigned Send Heartbeat timeout value. For example:

> An algorithm that finds a Minimum Propagation Time of 60 seconds and a Send Heartbeat timeout of 30 seconds will internally limit the Minimum Propagation Time period to 30 seconds. In this case, a non-changing, bound NVO will be propagated at the Send Heartbeat rate. A value that is changing faster than the internally limited propagation time period (30 seconds) will cause the object to propagate the value at the Send Heartbeat rate (also 30 seconds).

Start Up Delay Operation

During the NVO Start Up Delay timeout period, the NVO object’s algorithm monitors the Input and the Default input for a valid, changed value and updates the NVO object’s active value as required. Upon completion of the Start Up Delay timeout period, actual propagation of NVO data to the network is allowed and the NVO Send Heartbeat function’s timers are initiated.

When the Start Up Delay timeout is completed, an active value of a bound NVO that reflects a valid Input is actively propagated to the network. In the case where the active value reflects the Default input or the NA Value configuration property, propagation of the active value is based upon the setting of the NA Behavior configuration property. For structured SNVTs, all active values within the structure must reflect default conditions.

An NA Behavior setting of “Continue Heartbeat” causes the NVO object’s algorithm to propagate the default value due to a change of value or a Send Heartbeat timeout. An NA Behavior setting of “Interrupt Heartbeat” causes the algorithm to suspend network propagation of the default value regardless of a change of value or a Send Heartbeat timeout.

Reset Operation

Upon controller reset, the NVO object initializes the active value to the assigned “NA Value.” This applies to all resets, including controller power ups and software resets. NVO active values are not retained through a controller reset.

Engineering Units

Refer to Table–6.15 for a list of the metric, English, and optional conversion units that are assigned to SNVTs used with the MN 800 controller.

Available SNVTs

Table–6.14 Available LONMARK SNVTs for NVO Objects Copied from Network Variables Stencil.

Category	Available SNVTs		
Simple SNVTs	1 SNVT_amp	35 SNVT_speed_mil	71 SNVT_grammage
	2 SNVT_amp_mil	39 SNVT_temp	72 SNVT_grammage_f
	3 SNVT_angle	41 SNVT_vol	75 SNVT_freq_f
	4 SNVT_angle_vel	42 SNVT_vol_kilo	76 SNVT_freq_hz
	5 SNVT_btu_kilo	43 SNVT_vol_mil	77 SNVT_freq_kilohz
	6 SNVT_btu_mega	44 SNVT_volt	78 SNVT_freq_milhz
	7 SNVT_char_ascii	45 SNVT_volt_dbmv	79 SNVT_lux
	8 SNVT_count	46 SNVT_volt_kilo	81 SNVT_lev_percent
	9 SNVT_count_inc	47 SNVT_volt_mil	82 SNVT_multiplier
	13 SNVT_elec_kwh	48 SNVT_amp_f	98 SNVT_pwr_fact
	14 SNVT_elec_whr	49 SNVT_angle_f	99 SNVT_pwr_fact_f
	15 SNVT_flow	50 SNVT_angle_vel_f	100 SNVT_density
	16 SNVT_flow_mil	51 SNVT_count_f	101 SNVT_density_f
	17 SNVT_length	52 SNVT_count_inc_f	102 SNVT_rpm
	18 SNVT_length_kilo	53 SNVT_flow_f	104 SNVT_angle_deg
	19 SNVT_length_micr	54 SNVT_length_f	105 SNVT_temp_p
	20 SNVT_length_mil	55 SNVT_lev_cont_f	107 SNVT_time_sec
	21 SNVT_lev_cont	56 SNVT_mass_f	110 SNVT_area
	23 SNVT_mass	57 SNVT_power_f	113 SNVT_press_p
	24 SNVT_mass_kilo	58 SNVT_ppm_f	123 SNVT_time_min
	25 SNVT_mass_mega	59 SNVT_press_f	124 SNVT_time_hour
	26 SNVT_mass_mil	60 SNVT_res_f	125 SNVT_ph
	27 SNVT_power	61 SNVT_sound_db_f	126 SNVT_ph_f
	28 SNVT_power_kilo	62 SNVT_speed_f	129 SNVT_smo_obscur
	29 SNVT_ppm	63 SNVT_temp_f	131 SNVT_temp_ror
	30 SNVT_press	64 SNVT_time_f	138 SNVT_volt_ac
	31 SNVT_res	65 SNVT_vol_f	139 SNVT_amp_ac
	32 SNVT_res_kilo	66 SNVT_volt_f	143 SNVT_turbidity
	33 SNVT_sound_db	67 SNVT_btu_f	144 SNVT_turbidity_f
	34 SNVT_speed	68 SNVT_elec_whr_f	
Enumerated SNVTs	11 SNVT_date_day	103 SNVT_hvac_emerg	120 SNVT_defr_mode
	22 SNVT_lev_disc	108 SNVT_hvac_mode	121 SNVT_defr_term
	38 SNVT_telcom	109 SNVT_occupancy	122 SNVT_defr_state
	69 SNVT_config_src	118 SNVT_evap_state	145 SNVT_hvac_type
	97 SNVT_override	119 SNVT_therm_mode	
Structured SNVTs	83 SNVT_state	95 SNVT_switch	127 SNVT_chlr_status
	85 SNVT_zerospan	106 SNVT_temp_setpt	128 SNVT_tod_event
	87 SNVT_elapsed_time	111 SNVT_hvac_overid	
	91 SNVT_muldiv	112 SNVT_hvac_status	

Unit Conversion

Table–6.15 lists the metric, English, and optional conversion units that are assigned to SNVTs available within the MN 800 controller.

Table–6.15 Conversion Units Assigned to SNVTs Used in MN 800 Controllers.

SNVT #	SNVT Name	Element Name	Conversion Options				
			Default (based on controller units assigned)		Optional Units Available		
			Metric	English	2	3	4
1	SNVT_amp	—	amps	amps	—	—	—
2	SNVT_amp_mil	—	mA	mA	—	—	—
3	SNVT_angle	—	radians	radians	—	—	—
4	SNVT_angle_vel	—	radians / sec	radians / sec	rpm	—	—
5	SNVT_btu_kilo	—	kilo btu	kilo btu	—	—	—
6	SNVT_btu_mega	—	mega btu	mega btu	—	—	—
7	SNVT_char_ascii	—	—	—	—	—	—
8	SNVT_count	—	counts	counts	—	—	—
9	SNVT_count_inc	—	counts	counts	—	—	—
11	SNVT_date_day	—	day of week	day of week	—	—	—
13	SNVT_elec_kwh	—	kilowatt-hours	kilowatt-hours	—	—	—
14	SNVT_elec_whr	—	watt-hours	watt-hours	—	—	—
15	SNVT_flow	—	liters / sec	cfm	cfm	gallons / sec	gallons / min
16	SNVT_flow_mil	—	milliliters / sec	cfm	cfm	ounces / sec	—
17	SNVT_length	—	meters	feet	feet	—	—
18	SNVT_length_kilo	—	kilometers	miles	miles	—	—
19	SNVT_length_micr	—	micrometers	micrometers	—	—	—
20	SNVT_length_mil	—	millimeters	inches	inches	—	—
21	SNVT_lev_cont	—	percent	percent	—	—	—
22	SNVT_lev_disc	—	discrete level	discrete level	—	—	—
23	SNVT_mass	—	grams	ounces	ounces	pounds	—
24	SNVT_mass_kilo	—	kilograms	pounds	pounds	—	—
25	SNVT_mass_mega	—	metric tons	tons	tons	—	—
26	SNVT_mass_mil	—	milligrams	ounces	ounces	—	—
27	SNVT_power	—	watts	watts	btu / hour	tons refrigeration	horsepower
28	SNVT_power_kilo	—	kilowatts	kilowatts	—	—	—
29	SNVT_ppm	—	parts / million	parts / million	—	—	—
30	SNVT_press	—	kilopascals	inches WC	inches WC	psi	—
31	SNVT_res	—	ohms	ohms	—	—	—
32	SNVT_res_kilo	—	kilohms	kilohms	—	—	—
33	SNVT_sound_db	—	decibels	decibels	—	—	—
34	SNVT_speed	—	meters / sec	feet / sec	feet / sec	feet / min	mph
35	SNVT_speed_mil	—	meters / sec	feet / sec	feet / sec	feet / min	mph
36	SNVT_str_asc	—	—	—	—	—	—
38	SNVT_telcom	—	mode	mode	—	—	—
39	SNVT_temp	—	degrees C	degrees F	degrees F	degrees F (no offset)	—
41	SNVT_vol	—	liters	cubic feet	cubic feet	gallons	—
42	SNVT_vol_kilo	—	kiloliters	cubic feet	cubic feet	gallons	—
43	SNVT_vol_mil	—	milliliters	fluid ounces	fluid ounces	—	—

Table–6.15 Conversion Units Assigned to SNVTs Used in MN 800 Controllers. (Continued)

SNVT #	SNVT Name	Element Name	Conversion Options				
			Default (based on controller units assigned)		Optional Units Available		
			Metric	English	2	3	4
44	SNVT_volt	—	volts	volts	—	—	—
45	SNVT_volt_dbmv	—	dB uv	dB uv	—	—	—
46	SNVT_volt_kilo	—	kilovolts	kilovolts	—	—	—
47	SNVT_volt_mil	—	millivolts	millivolts	—	—	—
48	SNVT_amp_f	—	amps	amps	—	—	—
49	SNVT_angle_f	—	radians	radians	—	—	—
50	SNVT_angle_vel_f	—	radians / sec	radians / sec	rpm	—	—
51	SNVT_count_f	—	counts	counts	—	—	—
52	SNVT_count_inc_f	—	counts	counts	—	—	—
53	SNVT_flow_f	—	liters / sec	cfm	cfm	gallons / sec	gallons / min
54	SNVT_length_f	—	meters	feet	feet	—	—
55	SNVT_lev_cont_f	—	percent	percent	—	—	—
56	SNVT_mass_f	—	grams	ounces	ounces	pounds	—
57	SNVT_power_f	—	watts	watts	btu / hour	tons refrigeration	horsepower
58	SNVT_ppm_f	—	parts / million	parts / million	—	—	—
59	SNVT_press_f	—	pascals	inches WC	inches WC	psi	—
60	SNVT_res_f	—	ohms	ohms	—	—	—
61	SNVT_sound_db_f	—	decibels	decibels	—	—	—
62	SNVT_speed_f	—	meters / sec	feet / sec	feet / sec	feet / min	mph
63	SNVT_temp_f	—	degrees C	degrees F	degrees F	degrees F (no offset)	—
64	SNVT_time_f	—	seconds	seconds	—	—	—
65	SNVT_vol_f	—	liters	cubic feet	cubic feet	gallons	—
66	SNVT_volt_f	—	volts	volts	—	—	—
67	SNVT_btu_f	—	btu	btu	—	—	—
68	SNVT_elec_whr_f	—	watt-hours	watt-hours	—	—	—
69	SNVT_config_src	—	config source	config source	—	—	—
71	SNVT_grammage	—	gsm	gsm	—	—	—
72	SNVT_grammage_f	—	gsm	gsm	—	—	—
75	SNVT_freq_f	—	hertz	hertz	—	—	—
76	SNVT_freq_hz	—	hertz	hertz	—	—	—
77	SNVT_freq_kilohz	—	kilohertz	kilohertz	—	—	—
78	SNVT_freq_milhz	—	millihertz	millihertz	—	—	—
79	SNVT_lux	—	lux	lux	—	—	—
81	SNVT_lev_percent	—	percent	percent	—	—	—
82	SNVT_multiplier	—	—	—	—	—	—

Table–6.15 Conversion Units Assigned to SNVTs Used in MN 800 Controllers. (Continued)

SNVT #	SNVT Name	Element Name	Conversion Options				
			Default (based on controller units assigned)		Optional Units Available		
			Metric	English	2	3	4
83	SNVT_state	Bit 0	boolean	boolean	—	—	—
		Bit 1	boolean	boolean	—	—	—
		Bit 2	boolean	boolean	—	—	—
		Bit 3	boolean	boolean	—	—	—
		Bit 4	boolean	boolean	—	—	—
		Bit 5	boolean	boolean	—	—	—
		Bit 6	boolean	boolean	—	—	—
		Bit 7	boolean	boolean	—	—	—
		Bit 8	boolean	boolean	—	—	—
		Bit 9	boolean	boolean	—	—	—
		Bit 10	boolean	boolean	—	—	—
		Bit 11	boolean	boolean	—	—	—
		Bit 12	boolean	boolean	—	—	—
		Bit 13	boolean	boolean	—	—	—
		Bit 14	boolean	boolean	—	—	—
		Bit 15	boolean	boolean	—	—	—
84	SNVT_time_stamp	year	year	year	—	—	—
		month	month	month	—	—	—
		day	day	day	—	—	—
		hour	hours	hours	—	—	—
		minute	minutes	minutes	—	—	—
		second	seconds	seconds	—	—	—
85	SNVT_zerospan	zero	percent	percent	—	—	—
		span	—	—	—	—	—
87	SNVT_elapsed_time	day	days	days	—	—	—
		hour	hours	hours	—	—	—
		minute	minutes	minutes	—	—	—
		second	seconds	seconds	—	—	—
		millisecond	milliseconds	milliseconds	—	—	—
91	SNVT_muldiv	multiplier	—	—	—	—	—
		divisor	—	—	—	—	—
95	SNVT_switch	value	percent	percent	—	—	—
		state	—	—	—	—	—
97	SNVT_override	—	—	—	—	—	—
98	SNVT_pwr_fact	—	—	—	—	—	—
99	SNVT_pwr_fact_f	—	—	—	—	—	—
100	SNVT_density	—	kg / m^3	lbs / ft^3	lbs / ft^3	—	—
101	SNVT_density_f	—	kg / m^3	lbs / ft^3	lbs / ft^3	—	—
102	SNVT_rpm	—	rpm	rpm	—	—	—
103	SNVT_hvac_emerg	—	—	—	—	—	—
104	SNVT_angle_deg	—	degrees	degrees	—	—	—
105	SNVT_temp_p	—	degrees C	degrees F	degrees F	degrees F (no offset)	—

Table–6.15 Conversion Units Assigned to SNVTs Used in MN 800 Controllers. (Continued)

SNVT #	SNVT Name	Element Name	Conversion Options				
			Default (based on controller units assigned)		Optional Units Available		
			Metric	English	2	3	4
106	SNVT_temp_setpt	occupied cool	degrees C	degrees F	degrees F	degrees F (no offset)	—
		standby cool	degrees C	degrees F	degrees F	degrees F (no offset)	—
		unoccupied cool	degrees C	degrees F	degrees F	degrees F (no offset)	—
		occupied heat	degrees C	degrees F	degrees F	degrees F (no offset)	—
		standby heat	degrees C	degrees F	degrees F	degrees F (no offset)	—
		unoccupied heat	degrees C	degrees F	degrees F	degrees F (no offset)	—
107	SNVT_time_sec	—	seconds	seconds	—	—	—
108	SNVT_hvac_mode	—	—	—	—	—	—
109	SNVT_occupancy	—	—	—	—	—	—
110	SNVT_area	—	m^2	ft^2	ft^2	—	—
111	SNVT_hvac_overid	state	—	—	—	—	—
		percent	percent	percent	—	—	—
		flow	liters / sec	cfm	cfm	gallons / sec	gallons / min
112	SNVT_hvac_status	hvac mode	—	—	—	—	—
		primary heat	percent	percent	—	—	—
		secondary heat	percent	percent	—	—	—
		cooling	percent	percent	—	—	—
		economizer	percent	percent	—	—	—
		fan	percent	percent	—	—	—
		alarm	boolean	boolean	—	—	—
113	SNVT_press_p	—	pascals	inches WC	inches WC	psi	—
118	SNVT_evap_state	—	—	—	—	—	—
119	SNVT_therm_mode	—	—	—	—	—	—
120	SNVT_defr_mode	—	—	—	—	—	—
121	SNVT_defr_term	—	—	—	—	—	—
122	SNVT_defr_state	—	—	—	—	—	—
123	SNVT_time_min	—	minutes	minutes	—	—	—
124	SNVT_time_hour	—	hours	hours	—	—	—
125	SNVT_ph	—	ph	ph	—	—	—
126	SNVT_ph_f	—	ph	ph	—	—	—
127	SNVT_chlr_status	run mode	—	—	—	—	—
		operating mode	—	—	—	—	—
		chiller state - Bit 0 (in alarm)	boolean	boolean	—	—	—
		Bit 1 (run enabled)	boolean	boolean	—	—	—
		Bit 2 (local)	boolean	boolean	—	—	—
		Bit 3 (limited)	boolean	boolean	—	—	—
		Bit 4(chilled water flow)	boolean	boolean	—	—	—
		Bit 5(conditioned water flow)	boolean	boolean	—	—	—

Table–6.15 Conversion Units Assigned to SNVTs Used in MN 800 Controllers. (Continued)

SNVT #	SNVT Name	Element Name	Conversion Options				
			Default (based on controller units assigned)		Optional Units Available		
			Metric	English	2	3	4
128	SNVT_tod_event	current state	—	—	—	—	—
		next state	—	—	—	—	—
		time to next state	minutes	minutes	—	—	—
129	SNVT_smo_obscur	—	percent	percent	—	—	—
131	SNVT_temp_ror	—	deg C / min	deg F / min	deg F / min	—	—
138	SNVT_volt_ac	—	volts ac	volts ac	—	—	—
139	SNVT_amp_ac	—	amps ac	amps ac	—	—	—
143	SNVT_turbidity	—	NTU	NTU	—	—	—
144	SNVT_turbidity_f	—	NTU	NTU	—	—	—
145	SNVT_hvac_type	—	—	—	—	—	—

Network Management

LONWORKS is a peer-to-peer communications architecture that allows the direct sharing of data between networked I/A Series MicroNet controllers, as well as other LONWORKS devices. Data exchange is performed through the use of network variables (NVs), which are defined by a set of Standard Network Variable Types (SNVTs). Using network variables, MicroNet controllers can also exchange data directly with other manufacturers' LONWORKS controllers, providing a high level of interoperability.

Peer-to-peer "bindings" allow data to be exchanged between the network variable inputs (NVIs) and network variable outputs (NVOs) that reside in different devices. These bindings are engineered using a LONWORKS network management tool such as LONMAKER for Windows or I/A Series Niagara. In a general sense, the network management tool hides the details of the protocol and handles the overall installation process, based upon the project's needs. This includes assignment of the node's address, the creation of network variable bindings, the selection of appropriate delivery services (acknowledged, unacknowledged, unacknowledged repeated, etc.), the assignment of network variable selector IDs, and the sending of all messages required to configure the node's internal address tables. A network management tool also maintains a database that contains a record of all the node addresses, network variable connections, and default configuration properties.

Network Address

The network management tool assigns a network address, consisting of a domain, subnet, and node address, to each newly installed node. This process gives each node a totally unique address on the network. A device, or node, may belong to two domains, with each domain entry containing a subnet and a node address. In addition, each node may have up to fifteen group addresses. An authentication key can be generated for each domain entry, to ensure secure communications.

Network Variable Binding

Exchange of data between nodes is typically achieved through network variable *binding*. Network variable binding is the process by which network variable outputs (NVOs) on one or more nodes are connected to network variable inputs (NVIs) on other nodes. Bindings are only allowed between network variables with *matching SNVT types*. For example, an NVO of type SNVT_temp_p can not be bound to an NVI of type SNVT_temp, because although both represent temperature in degrees Celsius, there are differences in both range and resolution.

Bindings in which an NVO of one node (in this example, an I/A MicroNet controller) is connected to the NVI of another node (or nodes), are made using a LONWORKS network management tool such as LONMAKER for Windows. A typical node on a LON may have multiple connections, both inbound (NVI to NVO) and outbound (NVO to NVI). Each node records its connections in tables, and stores the binding information in its non-volatile memory area.

Network Variable Updates

An application update of a node's bound NVO typically causes the node to propagate the new value to corresponding NVIs on the receiving node(s). NVO propagation can also occur as a result of periodic Send Heartbeat timeouts.

To maintain data integrity and prevent flooding the network with communications traffic, mechanisms and timeouts such as *Send Heartbeat*, *Receive Heartbeat*, *Minimum Output*, and *Minimum Propagation* are provided. These configure the MicroNet controller's operation on the active network.

Send Heartbeat

The *Send Heartbeat*, in conjunction with the Receive Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). Accordingly, the Send Heartbeat mechanism is applied to the node's network variable outputs. The Send Heartbeat setting(s) determines the maximum time interval (in seconds) that is allowed before a non-changing "bound" NVO is sent on the active network. The Send Heartbeat function ensures that a value is automatically sent to update a receiving node or group of nodes on a periodic basis, even when that value remains unchanged for a length of time. For additional information on the Send Heartbeat function for the standard controllers, refer to the "Send Heartbeat Control" section in this chapter, on page 547. For the MN 800 controller, refer to the descriptions of the four independent "Send Heartbeat timers", starting on page 561.

Receive Heartbeat

The *Receive Heartbeat*, in conjunction with the Send Heartbeat, defines the mechanism used to ensure data integrity between two communicating devices (nodes). Accordingly, the Receive Heartbeat mechanism is applied to the node's network variable inputs. The Receive Heartbeat setting(s) determines the maximum time interval (in seconds) that is allowed before a non-updated NVI is defined as "Out of Service." This causes the appropriate programmed fallback to occur. For additional information on the Receive Heartbeat function for the standard controllers, refer to the "Receive Heartbeat Control" section in this chapter, on page 545. For the MN 800 controller, refer to the descriptions of the four independent "Receive Heartbeat timers", starting on page 559.

Minimum Output Time

The *Minimum Output Time* defines the minimum time interval (in seconds) between the sending of each propagated NVO on the active network. The controller applies this interval to all "bound" NVO's. This parameter can also be described as the final "filter" applied to all NVO updates sent from the controller. For additional information on the Minimum Output Time function for the standard controllers, refer to the "Minimum Out Time Control" section in this chapter, on page 547. For the MN 800 controller, refer to the description of the mandatory NCI object, Minimum Out Time, on page 559.

Caution: Disabling the Minimum Output Time or setting this interval to a relatively small time delay can cause a controller with rapidly updating NVOs to flood the network with communications traffic, adversely affecting the network's overall throughput and operation.

Minimum Propagation Time (MN 800)

The Minimum Propagation Time input defines the time interval (in seconds) that must expire before an NVO object is allowed to propagate a changed value to the task that handles the actual sending of variables on the network. This parameter can also be described as a "filter" that can be used to limit the amount of network traffic generated by a bound NVO. For additional information on the MN 800 controller's Minimum Propagation Time function, refer to page 558.

Caution: Disabling the Minimum Propagation Time or setting this interval to a relatively small time delay can cause a controller with a rapidly updating NVO object to flood the network with communications traffic, adversely affecting the network's overall throughput and operation.

Time Selection Guidelines

When selecting time values, a Send Heartbeat value should be chosen that will not interfere with the Minimum Output Time, and as a general rule, the Receive Heartbeat timeout value should be set to a rate that is at least four times the Send Heartbeat rate.

For example, a MicroNet MN 800 controller has been installed and configured with 10 bound NVOs and a Minimum Output Time value of 1 second. Based on these settings, the Send Heartbeat timeout value should be set so that it exceeds the following calculated minimum value:

Send Heartbeat (Min) = Number of bound NVOs X Min Output Time

Send Heartbeat (Min) = 10 X 1 Second

Send Heartbeat (Min) = 10 Seconds

The result of the above calculation shows that the assigned Send Heartbeat timeout value should not be set below 10 seconds. However, because a rate of 10 seconds is abnormally fast, a typical Send Heartbeat timeout of 120 seconds will be used, instead. Now that the Send Heartbeat timeout has been determined, calculate the Receive Heartbeat timeout as follows.

Receive Heartbeat $\geq$ Send Heartbeat X 4

Receive Heartbeat $\geq$ 120 seconds X 4

Receive Heartbeat $\geq$ 240 seconds (300 seconds is chosen)

In summary, then, Minimum Output Time is set to 1 second, Send Heartbeat is set to 120 seconds, and Receive Heartbeat is set to 300 seconds.

Configuration Properties

Configuration properties (NCIs) typically set up the internal workings of a device, as allowed by its manufacturer. These include setpoints, timing parameters, and tuning of algorithms, as well as other device performance items. NCI data is stored in the EEPROM of the controller and is retained through controller power cycles.

Caution: Because EEPROM supports only a finite number of write operations (approximately 10,000 write cycles), it is important that NCIs are not set up for frequent or continuous updates (changes). For this reason, bindings are not allowed between NVOs and NCIs.

Other Considerations

This section contains matters that require special attention when performing network management on a LON containing I/A Series Micronet controllers.

WP Tech (Download, Upload, Monitor, Browse) and Device Addresses

It must be remembered that WP Tech is *not* a LONWORKS network management tool. It is an engineering tool used to design, download, debug, and upload applications of I/A Series MicroNet controllers that are based on LONWORKS technology. WorkPlace Tech 4.0 is designed to comply with LonMark 3.3 requirements for certification as a passive tool when used with LonMark 3.3 certified controllers (Rev 4.1 standard controllers) and the MN 800.

As a passive tool WP Tech must not change the address table entries or the neuron state of a LonMark 3.3 certified controller or an MN 800. WP Tech requires that the controllers be configured and on-line in order for WP Tech to communicate with them. When a user initiates communications with a Rev 4.1 standard controller or an MN 800 controller that is not configured and on-line, WP Tech will respond as follows depending on the neuron state and the neuron mode of the controller:

1. Configured and on-line - normal state of operation, the red service LED on the controller is off and the controller will perform normal data exchange.
2. Configured and off-line - considered soft off-line, the service Led on the controller is off and the controller will discard any normal data exchange. WP Tech will notify the user that the controller must be set to on-line to proceed. If the user accepts, WP Tech will set the controller on-line and proceed. If the user chooses not to set the controller on-line the operation will be aborted. WorkPlace Tech will not set the controller back off-line after the operation is completed. (In the soft off-line state the controller will go on-line after a reset, so WP Tech is not required to return the controller off-line after the operation is completed).
3. Unconfigured - the service LED is flashing at a 1 second rate, WP Tech cannot determine if the controller's neuron mode is on-line or off-line. WP Tech will notify the user that the controller must be set configured and on-line to proceed. If the user accepts, WP Tech will configure the controller and set the controller on-line and proceed. If the user chooses not to configure the controller and set the controller on-line the operation will be aborted. WorkPlace Tech will return the controller to unconfigured and on-line after the operation is completed.
4. Configured, hard off-line - the service LED is off. With the neuron mode set to hard off-line, the mode will remain off-line through a reset. WP Tech will notify the user that the controller must be set to on-line to proceed. If the user accepts, WP Tech will set the controller on-line and proceed. If the user chooses not to set the controller on-line the operation will be aborted. WorkPlace Tech will set the controller back to a hard off-line after the operation is completed.
5. Applicationless - the service LED is on.The application is missing or corrupted.The neuron will not run in this state. WP Tech will notify the user that it cannot communicate with the controller. The controller is defective and must be replaced.

With this in mind, WP Tech may experience some difficulty when downloading to, uploading from, monitoring, or browsing for devices on the network. The success of these functions is directly related to the network topology and whether WP Tech is used before or after device addressing with a network management tool such as LONMAKER for Windows. The following scenarios describe these issues.

Flat LON Topology — Controllers not Yet Addressed with a Network Management Tool

In this scenario, the MicroNet controllers exist on a single segment or subnet that contains no routers. Additionally, these controllers have yet to be installed or addressed by a network management tool, and are still in the configured state in which they were received from the manufacturer, with the factory-assigned default network addresses. The factory-assigned addresses for MicroNet controllers are shown below, in Table–6.16.

Table–6.16 Factory-Assigned LONWORKS Addresses for MicroNet Controllers.

Address	Domain			Subnet	Node	Authentication
	Index	Size	Domain ID			
First	0	0	—	254	80	00 00 00 00 00 00
Second	Unused					

As Table–6.16 indicates, the first address is initialized to a zero length domain (see first domain index). WP Tech uses the zero length domain messaging as part of the Browse Network function, to identify and display a list of MicroNet devices on the network. Before the Download, Monitor, and Browse Network functions can operate, it may be necessary to reset the workstation network address to the default workstation address conditions, using the Workstation Addressing Wizard. For further details, see "Setting the Workstation Address to the Factory Default" section in Chapter 2 of this manual (page 43).

Note: Factory default network addresses are only temporary and remain active only until new addresses are assigned using the network management tool.

All Other Scenarios

This section covers all other scenarios, including:

- Flat LON topology in which controllers have already been addressed using a network management tool
- LON topology that includes routers and subnets already addressed using a network management tool
- Adding a new controller, yet to be addressed, to an existing network topology

In these scenarios, the controllers have been installed and addressed using a network management tool, and may exist on the near side or far side of a router.

For the Download, Upload, Monitor, and Browse Network functions to operate, the workstation network address must be synchronized to the domain and subnet of the LON network segment to which it is directly connected. This synchronization is performed using the Workstation Addressing Wizard. For further details, refer to the "Setting the Workstation Address Automatically" (page 43) or "Setting the Workstation Address Manually" section in Chapter 2 of this manual (page 43).

Note: WP Tech's Browse Network function will not properly detect and display a list of MicroNet controllers under the following conditions.

- Whenever a MicroNet controller's first and second domain addresses are *already assigned*. This can be corrected by synchronizing the WP Tech workstation to the LON network segment.
- Whenever a MicroNet controller's first and second domain addresses are *found to be "Unused" or unassigned*. This may occur if both domain addresses are set to "Unused" when the controller is uninstalled using a network management tool, or it may occur when the controller is set to an unconfigured state by pressing and holding the service pin. Synchronizing the WP Tech workstation to the LON network segment will not correct this situation. To perform a download under these conditions, the controller can be identified only by pressing its service pin or by manually entering its Neuron ID.

WP Tech Device Address Assignments

Regardless of the scenario, using WP Tech to download an application to or upload an application from a MicroNet controller will not affect the controller's assigned network address (domain, subnet, and node), as installed with a LONWORKS network management tool.

As a passive tool WP Tech must not change the address table entries or the neuron state of a LonMark 3.3 certified controller. WP Tech requires that the controllers be configured and on-line in order for WP Tech to communicate with them. When a user initiates communications with a controller that is not configured and on-line, WP Tech shall respond as described above (see "WP Tech (Download, Upload, Monitor, Browse) and Device Addresses")

WP Tech Real-time Data Monitoring and Point History Operation

WorkPlace Tech will not allow a user to perform monitoring of real-time data or perform point history operations on a Rev. 4.1 standard controller or MN 800 that is off-line, unconfigured, configured hard off-line or applicationless controller.

A Rev. 4.1 standard controller or MN 800 can be set configured and on-line using the configure device command in the Tools menu. After monitoring or point history operations are complete, the controller should be returned to its previous state using a network management tool or the Unconfigure Device command in the Tools menu.

Profile and Bindings Issues

Network management tools, such as LONMAKER for Windows, and I/A Series Niagara upload a MicroNet controller's profile and use it to perform controller-related network tasks such as bindings, property configuration changes, and other network functions. Changing the profile or network image of a controller, after its installation on the network, would cause the

controller and the network management tool to contain different and possibly mismatched network images, resulting in unpredictable and improper network operation.

MicroNet Standard Controllers

MicroNet standard controllers contain fixed, LONMARK-defined profiles. Because the profiles are fixed, WP Tech cannot be used to change a standard controller's network image, therefore it is not possible for the network images in the controller and the network management tool to become mismatched as a result of WP Tech downloads.

MicroNet MN 800 Controller

Caution: Downloading an application to a MicroNet MN 800 controller, by which the controller's network image or profile is changed, will clear the controller's internal network variable binding tables, causing improper or erratic operation on the network.

MicroNet MN 800 controllers contain flexible, user-definable profiles. This means that WP Tech can be used to customize the network image within the application. Since the profiles can be changed, downloading an application that includes an added or deleted SNVT object will cause a mismatch of network images and, more importantly, the external interface file (XIF). Also, the controller, upon recognizing the change in the SNVT tables, *will automatically clear* any existing *bindings* within the controller.

Downloading a profile that has been *modified* (i.e. contains added or deleted SNVT objects) *and not renamed* causes the controller to contain a new external interface file that cannot be properly handled by the network management tool. The user must delete the existing template from the network management tool, to force the tool to reread and update its XIF library where a copy of the project's network images is contained. An MN 800 controller that contains a modified profile should be renamed, especially if other controllers on the same network have previously used the same external interface file (XIF) profile name.

Note: Always rename the controller after making SNVT changes.

Mismatches can also result when creating a *new* application *within the same project or installation*. Creating multiple controller applications that contain different network images, but which have identical profile names, will cause a network management tool to use the *first* uploaded external interface file for *all subsequent installs* of controllers with the same profile name, causing a mismatch between the network image in the network management tool and those in the controllers. Unpredictable and improper network operation will occur.

LONWORKS Message Services

When using a network management tool to engineer a binding between devices, user options are often available for selecting the type of message services to be used. LONWORKS message service types for bindings include:

- Unacknowledged
- Unacknowledged, Repeated
- Acknowledged

Instead of specifying a particular message service when making a binding, some network management tools such as LONMAKER for Windows allow for the selection of a default message service (either “Auto” or “Application Configured”). The message used will be one of the LONWORKS types listed above.

MicroNet Controller Message Service Defaults

It is important to know that all I/A Series MicroNet controllers contain a default message type for network variable bindings. Depending on the controller and its firmware version, the default message types vary as follows.

MicroNet Standard Controllers

MicroNet standard controllers with Revision 3 and higher firmware contain default message types of “unacknowledged.”

MicroNet standard controllers with pre-Revision 3 firmware contain default message types of “user defined.” This typically defaults to the “acknowledged” message type when the default “Auto” or “Application Configured” message service is accepted for the binding.

MicroNet MN 800 Controllers

MN 800 controllers contain default message types of “unacknowledged.”

Caution: Use caution when using “acknowledged” message types because they require higher communications overhead. This is particularly true in the case of one-to-many, or group, addressing scenarios. In this case, an acknowledged message type causes much higher traffic and a reduction in overall communications throughput because one send requires responses from many receivers.

When using “acknowledged” message types, it is recommended that bindings be engineered using “Unacknowledged” or “Unacknowledged, Repeated” message services instead, especially for non-critical data. “Acknowledged” message service should be reserved for data that is critical, or where network traffic is not an issue.

Note: As a general rule, when engineering a binding with a network management tool, you should select the specific message service you need, and not accept a “default” message service type.

Other NV Data Exchange

Apart from “NV updates” between bound devices (NVO to NVI), NVs can service “data requests” from other devices. The requesting device is not “bound,” but receives NV information from “NV poll” or “NV fetch” operations. A PC running a LONWORKS program uses NV polling to extract data from networked LONWORKS devices. For example, most LON network management tools offer a “Browse” feature to examine real-time values in devices on the network. I/A Series Niagara also polls all values when in the Properties Screen. This causes additional communications overhead.

LNC-100 and NV Polling

The Invensys LON Network Controller (LNC-100) also uses NV polling (versus NV updates) to read and write NVs in LONWORKS devices on its LON bus. The LNC-100 itself is never “bound” to any other LONWORKS devices.

Instead, it must be programmed using NETWORK 8000 type SIM “blocks,” specially configured for accessing particular NVs in devices on the network. This process is called SIM-block “mapping” of NVs.

Note: Devices required to be polled by the LNC-100 must first be installed (addressed) by a LON network management tool before SIM-block mapping is done. For details on LNC-100 programing, including the appropriate engineering tools, refer to the *LON Network Controller Engineering Guide*, F-26566.

NV polling is described here because it is an important consideration when building an application in a MicroNet controller, particularly if data items in the application need to be accessed from an LNC-100. These data items should be connected to NVs (NCIs, NVIs, and NVOs) in the controller’s profile.

Once in the LNC-100, values from LONWORKS NVs can be shared with non-LONWORKS controllers on the LNC’s ASD bus. If the LNC-100 is equipped with a NIM, the LONWORKS NVs can also be shared with other LNCs, GCMs, or DMS panels on the NIM network. These NV values are then also accessible in the various NETWORK 8000 user interfaces or HMIs, such as InVue, Signal, and UltiVist.

Point History Log

Point History Logging is a standard feature included in each MN 800 controller. It provides data that can be uploaded into WP Tech for use during initial setup and checkout of a control system. This data may also be used when servicing a control system that is experiencing control or operational difficulty.

Tracked Data

The Point History Logging function allows the tracking of up to 24 individual analog or digital points. Historical data is maintained in backed up memory, which retains previously collected data through controller power cycles (power-downs and resets). Both analog data and digital data include a time stamp that indicates the time at which the sample was collected (analog) or the change of state occurred (digital).

The most recent 48 samples or changes of state for each point are maintained in the log, and the data for each point within the log rotates when full. If *fewer than* 48 samples have been collected since the previous upload, the system compares data and removes the overlapping data, then appends only the new data to the previous, with no blank spaces or special notations between the uploads. If *more than* 48 samples have been collected since the previous upload, this means that some data has been rolled over and is no longer in the database. In this event, the system knows that there is a gap in the collected data, and it indicates this gap by inserting asterisks in the date, value, and units fields in the report row, between the two collection uploads.

Data Tracking Pens

The history log provides 24 tracking pens, each of which can be set up as analog or digital, and can be assigned to any point within the application, including object outputs, S-Link tags, etc.

Analog Tracking Pens

Tracking data logged by an analog tracking pen can show changes (i.e. temperature, humidity, object outputs, etc.) in the control system over a period of time. Analog tracking pens are assigned a time value that defines the frequency of data sampling and collection. This collection frequency is defined through the sample units, which include hour, minute, and second, and the sample rate, which ranges between 1 and 1000. This means an analog tracking pen can be set to collect data at any single frequency, from the minimum of 1 second, up to the maximum of 1000 hours. A running log of the last 48 samples (including time stamps) is maintained.

Digital Tracking Pens

Digital tracking pens compare new digital values to previous digital values, and log all changes of state when they occur. Digital states include Digital OFF, Digital ON, and "not active" (NA), and the minimum rate for change of state is 1 second. Data logged by a digital tracking pen can inform the user as to the frequency at which certain equipment has been turned OFF and ON. A running log of the last 48 changes of state (including time stamps) is maintained.

Data Format

The data collected for both analog and digital tracking is stored by utilizing the same basic data / time stamp format. In both analog and digital data, the time stamp contains year, month, day, hour, minute, and second information. A controller whose time has not been set or synchronized, will contain a time stamp that reflects only a 24 hour time period (hours, minutes, and seconds). In this situation, the time stamp is for reference only, and the date portion of the time stamp (month, day, and year) will indicate a not active (NA / NA / NA).

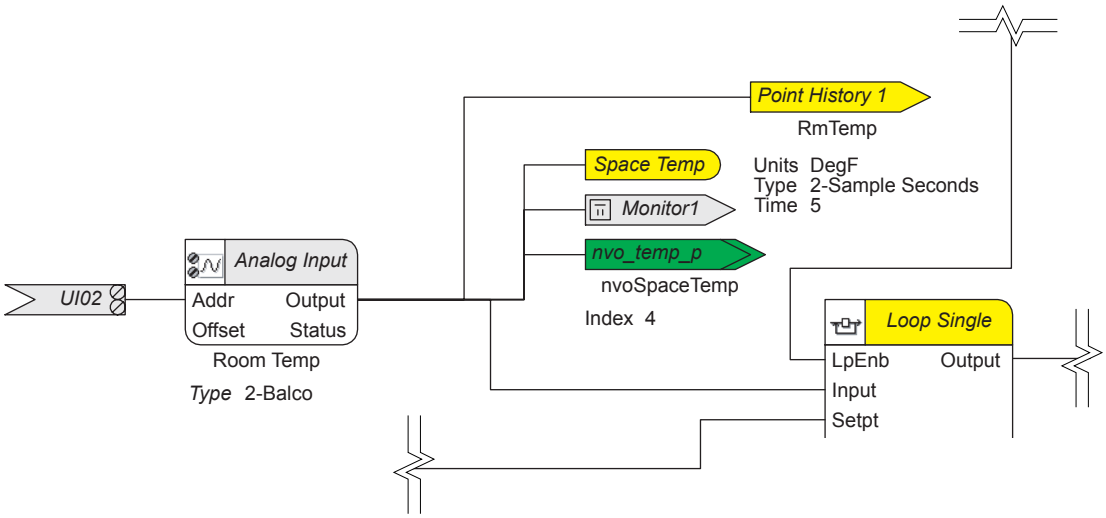
Application

The Point History Log function is applied by adding a point history tag to an application drawing. To do this, click and drag a point history tag from a stencil, to the desired location within the application drawing Figure-6.17. The Select Object Type dialog box appears, listing the available (unused) tracking pens, which are labeled “Point History 1” through “Point History 24”. Once a tracking pen is selected, WP Tech places a Point History tag at the location of the drop.

Once the Point History tag has been copied onto the drawing page, the user connects it to the selected output. Then, the user right-clicks on the tag to open the shortcut menu and selects **Customize.** This opens the Customize Object window, displaying the point history tag’s Object Name. The default name is “Histx,” where “x” denotes the selected tracking pen number. This name may be changed as needed.

Note: Each point history tag’s Object Name and associated tracking pen number will be placed in the comma separated values file (CSV) whenever the point history data is uploaded using the WP Tech “Point History Upload” feature.

Clicking on the Properties tab in the Customize Object window displays a list of properties which can be selected for display below the point history tag. No properties are displayed in the default condition.



Figure–6.17 Point History Tag Used to Sample Output from Analog Input Object.

The user then right-clicks the point history tag and chooses **Properties** to bring up the properties editor, which displays the configuration properties for the point history tag. WP Tech defaults point history tags in the “Disabled” condition, meaning the user must configure the tag for Analog or Digital use. Refer to Table–6.17 and Table–6.18 for detailed descriptions of the available properties of point history tags.

Properties

Table–6.17 Point History Tag Configuration Properties.

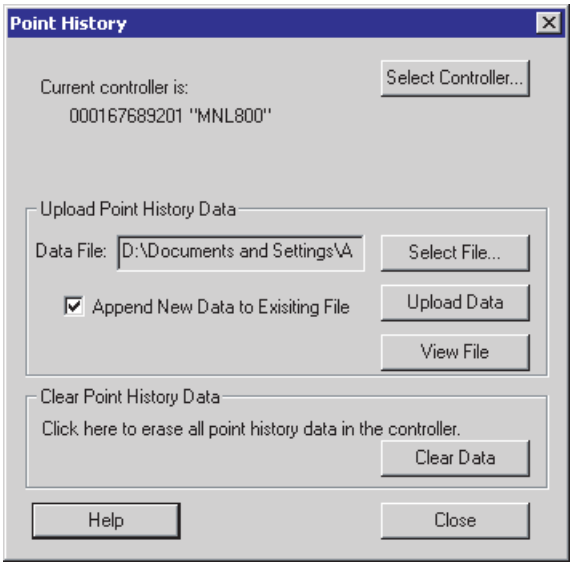
Name	Description	Default	Range/Select
Description	Character string. Optional user-defined descriptor that may be used to further describe the point history tag.	—	Up to 250 characters
Units	Units for the data values generated by this point. The units entered in this string will be placed together with the measured point data, in the comma separated values file (CSV), during the upload of point history data using the WP Tech “Point History Upload” feature. Only the first eight characters of the Units string will appear in the CSV file.	—	—
Type	Sample Type, used to identify the input as a digital value, or used in conjunction with the Sample Time to set the sample rate.	0	0 - Disabled 1 - Digital 2 - Sample Seconds 3 - Sample Minutes 4 - Sample Hours
Sample Time	Integer value. The sample time, in seconds, minutes, or hours as defined by Type. The sample time is ignored by the point history pen whenever Type is set to “Disabled” or “Digital.”	—	1 to 1000

Table–6.18 Point History Tag Input Properties.

Name	Description	Range/Select
Track Value	The sampled analog or digital value.	Range and selection are determined at the point being sampled.

Accessing Collected Data

The collected point history data, consisting of data and associated time stamps, is stored in the controller. This data can be uploaded into WP Tech by clicking **Application** in the main menu, then **Point History Upload**. The Point History dialog box appears Figure-6.18.



Figure–6.18 Point History Dialog Box.

Clicking the Upload Data button causes WP Tech to upload the data from the controller, into a comma delimited file (CSV). This file can then be imported into a spreadsheet program such as Excel, where it can be charted and graphed. For detailed instructions on accessing point history data, refer to the *WorkPlace Tech Tool 4.0 User's Guide*, F-27255.

Service Pin Operation

Each I/A Series MicroNet controller has an onboard Service Pin pushbutton switch. This pushbutton is typically used to select and identify the controller when downloading an application using WP Tech, or when addressing the controller with a network management tool such as LONMAKER for Windows. Each time the Service Pin is momentarily pressed, the controller broadcasts, over the LON, a Service Pin message that contains the controller's Program ID and Neuron ID. The Program ID identifies the controller's network image, or profile. The Neuron ID is a unique identifier that is given to every Neuron device produced. A Service Pin message can be sent as frequently as required, as it has no lasting effect other than to send a simple broadcast message.

MicroNet MN 800 controllers and all MicroNet standard controllers with Revision 3 and higher firmware perform one additional Service Pin function. Pressing and holding the Service Pin pushbutton *for 6 seconds* causes the controller to go into an *unconfigured state*, in which the Service LED flashes at a rate of 1 second On / 1 second Off. *More importantly*, this *resets* all the controller's assigned LON addresses to "Unused" and clears the assigned authentication key.

This feature is specifically designed to allow a controller to be "reclaimed." This may be necessary, for example, when a network management tool has been used in such a way as to inadvertently prevent WP Tech's access to a controller. Although unlikely, this scenario could occur on a multiple-vendor jobsite where message authentication is used.

Note:

- If the MicroNet controller has an attached MN-S2, S3, S4, or S5 sensor, an alternate (and often more convenient) method exists for sending a Service Pin message. This method consists of pressing and holding the sensor's Override button for a period of 8 seconds or longer. Likewise, a Service Pin message can be produced with an MN-S4-FCS sensor, by pressing and holding its Fan Off / On button for 8 seconds or longer.
- MicroNet standard controllers suspend processing of the downloaded application when in an unconfigured state.

Diagnostic Indicators

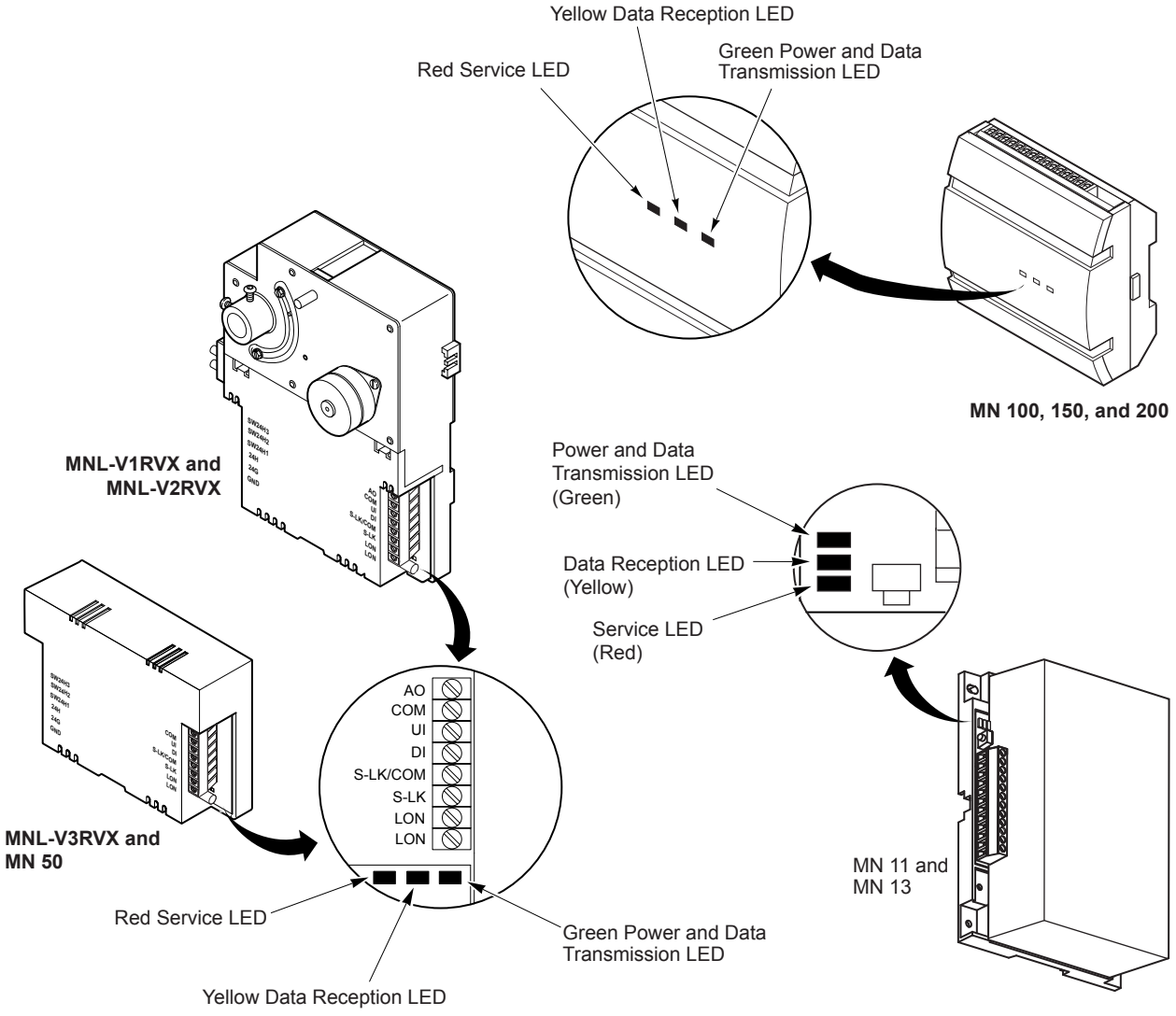
I/A Series MicroNet controllers feature LED indicators. The LCD screen of an S-Link sensor can display error and alarm indications for the MicroNet controller to which it is connected.

MicroNet Controller LEDs

The I/A Series MicroNet controllers have three LEDs that represent power, communications activity and application status.

MN 50, 100, 110, 130, 150, 200, and VAV Series

See Figure–6.19 for the location of the LED indicators on MicroNet standard controllers. Refer to Table–6.19 to identify the LED indications and to determine any corrective action that may be required.



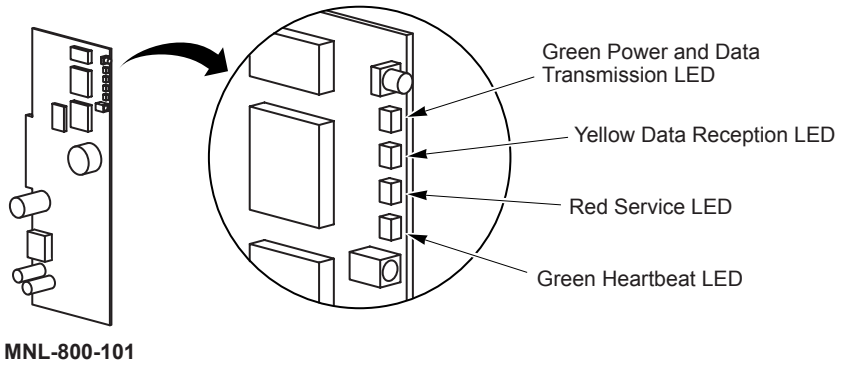
Figure–6.19 Locations of LEDs on MicroNet Standard Controllers.

Table–6.19 LED Indications on MicroNet Standard Controllers.

Indicator	Status	Corrective Action
Yellow LED – Data Reception	Blinks when the controller receives data from the LON.	None required.
	ON indicates one LONWORKS network wire may be disconnected.	Check the connections on all nodes.
	OFF indicates that data reception is not taking place.	None Required.
Green LED –Data Transmission	Blinks when the controller transmits data to the LON.	None Required.
	ON indicates that the controller is not transmitting data. ON also indicates that power is being applied to the controller.	
	OFF indicates no power to controller.	Check power.
Red LED – Service	OFF indicates the Neuron is operating properly, allowing normal communications to and from the HVAC application.	None required.
	OFF indicates the Neuron is off-line. The Neuron responds to network management messages only. In this state, communication to and from the HVAC application is not possible.	Place the Neuron on-line by downloading an application using WP Tech or configure the Neuron using a third party management tool.
	Blinking at a rate of 0.5 Hz (1 second ON, 1 second OFF) indicates that the Neuron application is loaded, but the Neuron's communication parameters are not loaded, are being reloaded, or have been corrupted. The Neuron is considered unconfigured. The Neuron's communication parameters cannot be configured by field personnel.	Use WP Tech to download the appropriate application. If the red service LED continues to blink, download the application two to three more times. If the red LED is still blinking, replace the controller. Or, recommission the controller with a network management tool. ***Note:*** Downloading an application using WP Tech will configure and place the Neuron on-line before the actual download. Existing subnet/node addresses will not be affected by WP Tech.
	Wink mode provides a visual means for identifying the controller, using WP Tech or a third party network management tool. During this mode, the LED blinks as follows: • With revision 3.x firmware or earlier: the Service LED blinks 1 second ON, 1 second OFF; cycles five times for a total of 10 seconds of wink time. • With revision 4.x firmware or later: the Service LED blinks 3 seconds ON, 1 second OFF; cycles three times for total of 12 seconds wink time.	None required.
	ON indicates that the Neuron application is not running. Neuron applications are not field replaceable.	Replace the controller.

MN 800 Series

See Figure–6.20 for the location of the LED indicators on MicroNet MN 800 controllers. Refer to Table–6.20 to identify the LED indications and to determine any corrective action that may be required.



Figure–6.20 Location of LEDs on MicroNet MN 800 Controller.

Table–6.20 LED Indications on MicroNet MN 800 Controllers.

Indicator	Status	Corrective Action
Yellow LED – Data Reception	Blinks when the controller receives data from the LON.	None required.
	ON indicates one LONWORKS network wire may be disconnected	Check connections on all nodes.
	OFF indicates that data reception is not taking place.	None Required.
Green LED – Data Transmission	Blinks when the controller transmits data to the LON.	None Required.
	ON indicates that the controller is not transmitting data. ON also indicates that power is being applied to the controller.	
	OFF indicates no power to controller.	Check power.
Red LED – Service	OFF indicates the Neuron is operating properly, allowing normal communications to and from the HVAC application.	None required.
	OFF indicates the Neuron is off-line. The Neuron responds to network management messages only. In this state, communication to and from the HVAC application is not possible.	Place the Neuron on-line by downloading an application using WP Tech or configure the Neuron using a third party management tool.
	Blinking at a rate of 0.5 Hz (1 second ON, 1 second OFF) indicates that the Neuron application is loaded, but the Neuron's communication parameters are not loaded, are being reloaded, or have been corrupted. The Neuron is considered unconfigured. The Neuron's communication parameters cannot be configured by field personnel.	Use WP Tech to download a "blank" application to clear the MN 800. Then, download the appropriate application. If the red service LED continues to blink, download the application two to three more times. If the red LED is still blinking, replace the controller. ***Note:*** Downloading an application using WP Tech will configure and place the Neuron on-line before the actual download. Existing subnet/node addresses will not be affected by WP Tech.
	ON indicates that the Neuron application is not running. Neuron applications are not field replaceable.	Replace the controller.

Table–6.20 LED Indications on MicroNet MN 800 Controllers. (Continued)

Indicator	Status	Corrective Action
Green LED – Heartbeat	Blinking at 0.5 second ON, 0.5 second OFF indicates the controller is operating normally.	None required.
	Wink mode provides a visual means for identifying the controller, using WP Tech or a third party network management tool. During this mode, the LED blinks as follows • With revision 1.x firmware: the Heartbeat LED blinks 2 seconds ON, .5 second OFF; cycles five times for a total of 12.5 seconds wink time. • With revision 2.0 firmware or later: the Heartbeat LED blinks 3 seconds ON, 1 second OFF; cycles three times for total of 12 seconds wink time.	None required.
	A diagnostic blink pattern of two quick flashes followed by a pause, indicating RAM/ROM failure.	Turn the power OFF, then ON. Replace the controller if necessary.
	A diagnostic blink pattern of three quick flashes followed by a pause, indicating ROM failure.	
	A diagnostic blink pattern of four quick flashes followed by a pause, indicating RAM failure.	
	OFF indicates the controller is not operating properly.	Check the power to the controller. Replace the controller if necessary.

S-Link Sensor Display

MicroNet MN-S3xx, MN-S4xx, MN-S4xx-FCS, and MN-S5xx S-Link sensors feature a Diagnostic Mode whereby the sensor's LCD screen can display a variety of diagnostic data including Error and Alarm indications and controller wink mode for the MicroNet controller to which it is connected.

Controller wink mode indication is only available with S-Link sensors interfacing with revision 4.x (or later) firmware on MicroNet standard controllers or revision 2.0 (or later) firmware on MNL-800 controllers. While a controller is in wink mode the red occupancy LED on the sensor blinks 1/2 second ON, 1/2 second OFF.

The MN-S4xx-FCS sensor does not have a red occupancy LED and, therefore, does not indicate when the controller is in wink mode.

Note: If the S-Link sensor is left idle for 30 seconds while in the Diagnostic Mode, the sensor times out and returns to the default screen.

To Access Error and Alarm Indications

To view the Errors and Alarms:

1. Press and hold both ends of the Up/Down Key for 4 seconds. The display automatically alternates between the controller's Subnet (Adr) and Node (nod) address.
2. Press the Up/Down Key to scroll through the Errors frame and the Alarms frame.

Error Indications

The Error frames display a value of 1, 2, 4, 8, 16, 32, or 64, or a combination of two or more of these values (error code 64 is used with MN 800 only). Each of the values, 1, 2, 4, 8, 16, 32, and 64, represents an error, as described in Table–6.21. When more than one error is present, the sum of the values associated with these errors is displayed instead. The LCD alternates between the Error frame and the numerical error value. An error screen displaying dashes (---) indicates no errors.

Example 1: An Error frame displays a value of 17. Because this value is not 1, 2, 4, 8, 16, 32, or 64, it must be the sum of one or more of these. The only possible combination is 1 and 16 (1 + 16 = 17). Therefore, the value “17” represents two errors, 1 = EEPROM write or read error, and 16 = Analog output writing error.

Example 2: An Error frame displays a value of 11. As with Example 1, this value is not 1, 2, 4, 8, 16, 32, or 64, so it must be a sum. The only possible combination is 1, 2, and 8 (1 + 2 + 8 = 11). Therefore, the value “11” represents three errors, 1 = EEPROM write or read error, 2 = Out of range Universal Input (UI1), and 8 = Low velocity pressure (the sensor is connected to an MNL-V2RVX controller).

Table–6.21 Error Code Descriptions.

Error Code	Controller		
	MNL-5RXX MNL-10RXX MNL-11RXX MNL-13RXX MNL-15RXX MNL-20RXX	**MNL-V1RVX MNL-V2RVX MNL-V3RVX**	**MN 800**
1	EEPROM write or read error		
2	Out of range universal input (UI1)		Out of range UI1 - UI8
4	Out of range universal input (UI2)	High velocity pressure	Out of range UI9 - UI6 (future use)
8	Out of range universal input (UI3)	Low velocity pressure	Out of range UI17 - UI24 (future use)
16	Analog output writing error		Not applicable
32	Calibration data checksum error or unit is uncalibrated		
64	Not applicable		Invalid RTC time

Alarm Indications

The Alarm frames display the last four alarms from the connected MicroNet controller. The LCD alternates between Alarm frame and the numerical alarm value. If the controller is sending more than one alarm, the numerical alarm value will update every 4 to 5 seconds. Alarms are defined by the controller application. For information regarding specific alarm definitions, refer to the “Analog Alarm” (page 114) and “Binary Alarm” (page 154) sections in Chapter 5 of this manual.

SNVT-Based Indicators

For MicroNet standard controllers with firmware revision 4.x or later, the nvoDeviceInfo SNVT provides the following diagnostic information:

- Model Number: 9 characters
- Firmware Version: 3 characters
- S-Link Sensor Diagnostic Status: 3 characters
- Application Name: 12 characters

Any tool that can read nvoDeviceInfo can display the information above. The S-Link Sensor diagnostic status, available within nvoDeviceInfo, uses the same error codes listed in Table–6.21 with the addition of Error Code 128. Error Code 128 indicates a non-communicating S-Link Sensor.

Appendix A
Memory Requirements

This chapter explains the memory usage of control objects in I/A Series MicroNet standard controllers and MN 800 controllers, including an alphabetical listing of all objects with their corresponding memory requirements.

Memory Types

Each control object created in a MicroNet controller consumes specific amounts of controller memory. The two types of memory are:

- EEPROM - Electrically Erasable Programmable Read Only Memory.
- RAM - Random Access Memory.

EEPROM is non-volatile memory, meaning it does not require backup when power is removed from the controller. EEPROM is used to store a copy of each control object, including all input properties, input pointers, and critical configuration properties. LONMARK profile items are also stored in EEPROM (copied from ROM), along with other MN sensor and controller configuration data. These are considered “fixed overhead items.” Once an application has been downloaded into an I/A Series MicroNet controller, it remains in EEPROM until changed from a programming tool, even if power is removed for long periods of time.

RAM is used as “operation memory.” On power-up or reset, most of the information stored in EEPROM is “shadowed” in RAM. However, not only does RAM contain a copy of each control object, but also continuously stores each object’s data results—that is, all the intermediate and output values in real time. This is why most types of control objects consume more RAM than EEPROM. This also means when calculating controller memory usage, RAM is more critical than EEPROM.

Controller Capacities

Standard Controller Memory

Each MicroNet standard controller has 2K bytes (2048 bytes) total of both EEPROM and RAM for all control objects plus fixed overhead items. Depending on profile type, the amount of available RAM (in bytes) for all control objects differs slightly among the various models of controllers. Refer to Table–A.1. MicroNet standard controllers also have EEPROM dedicated to storing drawing information. This drawing information is used by WorkPlace Tech to generate application drawings from uploaded controller information. Depending on the controller firmware rev, the amount of available EEPROM dedicated to drawing information varies.

Table–A.1 Standard Controller Memory, by Model/Profile.

Controller Models	Firmware Rev	Controller Profile	RAM Available for Control Objects	EEPROM Dedicated for Drawing Information
MNL-10RF1, -20RF1,	Pre-Rev3	Fan Coil	1668 bytes	2048 bytes (2 KB)
MNL-5RF2, -10RF2, -11RF2, -15RF2, -20RF2	Rev3 or Higher		1676 bytes	
MNL-5RF3, -10RF3, -11RF2, -13RF3, -15RF3, -20RF3	Rev4.1 or Higher			6168 bytes (6 KB)
MNL-10RH1, -20RH1,	Pre-Rev3	Heat Pump	1686 bytes	2048 bytes (2 KB)
MNL-5RH2, -10RH2, -15RH2, -20RH2	Rev3 or Higher		1694 bytes	
MNL-5RH3, -10RH3, -15RH3, -20RH3	Rev4.1 or Higher			6168 bytes (6 KB)
MNL-10RR1, -20RR1,	Pre-Rev3	Roof Top	1678 bytes	2048 bytes (2 KB)
MNL-5RR2, -10RR2, -15RR2, -20RR2	Rev3 or Higher		1686 bytes	
MNL-5RR3, -10RR3, -15RR3, -20RR3	Rev4.1 or Higher			6168 bytes (6 KB)
MNL-5RS1, -10RS1, -15RS1, -20RS1	Rev3 or Higher	Satellite 1 (Roof Top variation)	1598 bytes	2048 bytes (2 KB)
MNL-5RS2, -10RS2, -15RS2, -20RS2	Rev3 or Higher	Satellite 2 (Roof Top variation)	1590 bytes	
MNL-5RS3, -10RS3, -15RS3, -20RS3	Rev4.1 or Higher	Satellite 3 (Roof Top variation)	1598 bytes	6168 bytes (6 KB)
MNL-5RS4, -10RS4, -15RS4, -20RS4	Rev4.1 or Higher	Satellite 4 (Roof Top variation)	1590 bytes	
MNL-V1RV1, -V2RV1, -V3RV1	Pre-Rev3	VAV	1650 bytes	2048 bytes (2 KB)
MNL-V1RV2, -V2RV2, -V3RV2	Rev3 or Higher		1658 bytes	
MNL-V1RV3, -V2RV3, -V3R32	Rev4.1 or Higher			6168 bytes (6 KB)

MN 800 Memory

Each MN 800 controller has approximately 32K bytes of total database memory that is used to store data that includes control objects, the controller's network image, and other fixed memory "overhead" items. This means that there are, respectively, 15,350 bytes of EEPROM and 15,800 bytes of RAM available Table–A.2 for application control logic and network interface (SNVT object) use. EEPROM that is not used by overhead items and control objects is available storage of drawing information.

Table–A.2 MN 800 Controller EEPROM and RAM for Control Objects and SNVT Objects.

Model	Controller Profile	EEPROM Available	RAM Available
MNL-800-101	Programmable	15,350 bytes	15,800 bytes

Note: MicroNet sensor support is included in every I/A Series MicroNet controller's fixed overhead. This means selection of any MicroNet sensor model in an application (including none) makes no difference in available RAM.

The MN 800 and Rev.3 and higher standard controllers also include support for a controller schedule as part of the fixed overhead. No additional controller memory is used.

Object Memory Requirements

Each control object, when created, requires some controller RAM and EEPROM to exist and to perform its calculations. These memory requirements are listed in each object's description.

For example, the Analog Alarm object has these requirements:

Memory Requirements: (per object)

EEPROM: 22 bytes
RAM: 30 bytes (LONMARK) 8 bytes (MN 800)

Note: Several control objects are available only if programming a controller with Rev.3 or later firmware. Device support requirements are included in the object descriptions provided in Chapter 5.

WP Tech Statistics Function

WP Tech provides a Statistics function for checking the amount of available memory in any application. This information is based on the number and types of control objects currently in the application's drawing. Memory statistics can be compared to individual control object memory requirements listed in each object's description in Chapter 5, or to the memory requirements for each object type listed in the following table Figure-3.

For more details on running the applications Statistics function, refer to the *I/A Series WorkPlace Tech Tool 4.0 User's Guide*, F-27255.

Memory Requirements Per Object

Table–A.3 lists each control object type and the respective memory required. Objects listed in **boldface** are available only in Rev.3 MicroNet standard controllers or MicroNet MN 800 controllers, or both (i.e. these objects are not available in pre-Rev.3 LONMARK controllers).

Table–A.3 Memory Requirements for Control Objects.

Object Type	EEPROM (bytes)	RAM (bytes)	
	Standard and MN 800 Controllers	Standard Controllers	MN 800 Controllers
Abs Sub / Div	10	12	2
Add / Add	10	12	2
Add / Div	10	12	2
Analog Alarm	22	30	8
Analog Input	20	24	4
Analog Output	16	18	2
Analog Output Priority	22	26	4
AND / AND	10	12	2
AND / OR	10	12	2
Average	10	12	2
Binary Alarm	16	22	6
Binary Encoder	16	22	6
Binary Input	12	22	10
Binary Output	10	12	2
Calendar[a]	142	—	8
Clocked SR	10	14	4
Compare	10	12	2
Compare 2[b]	10	14	4
Control Override	14	20	6
Count Down	10	16	6
Count Up	10	16	6
COV Priority[b]	16	24	8
Curve Fit[b c]	34	38	4
Demux Select[b]	20	32	12
Dual Delay	12	20	8
Dual Minimum	12	20	8
DUI Expander[b c]	18	30	12
Enthalpy[a]	20	—	14
Event Indicator	14	20	6
EXOR	8	10	2
Fan Speed[c]	26	42	16
Filter	8	10	2
Floating Actuator	24	40	16
Floating Actuator Priority	30	48	18
High Select	10	12	2
Interlock	12	28	16
Interstage Delay (3)	26	44	18
Interstage Delay (6)	44	74	30
Interstage Delay (10)	68	114	46
Latch	10	14	4

Table–A.3 Memory Requirements for Control Objects. (Continued)

Object Type	EEPROM (bytes)	RAM (bytes)	
	Standard and MN 800 Controllers	Standard Controllers	MN 800 Controllers
Limit	10	12	2
Limit Thermostat[b]	10	12	2
Loop Sequenced	36	56	20
Loop Single	22	38	16
Low Select	10	12	2
MA Volume	10	12	2
Minimum Off	10	18	8
Minimum On	10	18	8
Momentary Start / Stop	16	26	10
Mul / Add	10	12	2
Mul / Div	10	12	2
Off Delay	10	18	8
On Delay	10	18	8
OR / AND	10	12	2
OR / OR	10	12	2
OSS[a]	52	—	82
Pressure Transducer[d]	10	18	—
Priority Input (2)	8	10	2
Priority Input (4)	12	16	4
Priority Value Select	22	24	2
PWM	18	24	6
PWM Priority	24	32	8
Ramp[a]	22	—	10
Reset	18	20	2
Schedule 7-Day[a]	206	—	14
Select	10	12	2
Sensor Input[a]	16	—	14
Sequence (3)	12	20	8
Sequence (6)	12	26	14
Sequence (10)	12	34	22
Setpoint Control	20	30	10
SqRt Mul / Add	10	12	2
SR Flip-Flop	8	10	2
Step Driver[a]	32	—	14
Sub / Add	10	12	2
Sub / Div	10	12	2
Sub / Mul	10	12	2
Sub / Sub	10	12	2
Thermostat	10	14	4
Thermostat 2[b]	12	16	4
VAV Actuator[d]	22	40	—

a.Available only in MN 800 controllers.
b.Available only in standard controllers with Rev.3 or later firmware and the MN 800.
c.Not available in LONMARK VAV controllers.
d.Available only in LONMARK VAV controllers.

Appendix B
Profiles and Network Data

This chapter provides quick reference information on each of the following HVAC LONMARK Functional Profiles as implemented in I/A Series LONMARK MicroNet controllers (profiles are listed with LONMARK profile number):

- Fan Coil Profile (8020).
- Heat Pump Profile (8051).
- Roof Top Unit Profile (8030).
- VAV Controller Profile (8010).
- Satellite 1 Profile (8030, Roof Top Unit variation 1).
- Satellite 2 Profile (8030, Roof Top Unit variation 2).
- Satellite 3 Profile (8030, Roof Top Unit variation 1).
- Satellite 4 Profile (8030, Roof Top Unit variation 2).

In addition, this chapter contains a brief description of the MN 800 User-Definable Profile.

This chapter also contains the following SNVT-related topics:

- SNVT to MicroNet Control Logic Data Exchange.
- MN Controllers SNVTs Quick Reference.
- Invensys-defined Network Variable Types.

Additional LONMARK details on all the various functional profiles and SNVTs can be acquired directly from the web site of the LONMARK Association in the form of 'User Guides' at the following URL:

http://www.LONMARK.org/

Each LONMARK profile document explains the behavior specified by compliance to the specific LONMARK profile. Other documents provide SNVT details, including *The SNVT Master List and Programmer's Guide*. Most documents are available as PDF files, which can be read using the widely-distributed Adobe Acrobat Reader program.

Invensys LONMARK Profiles

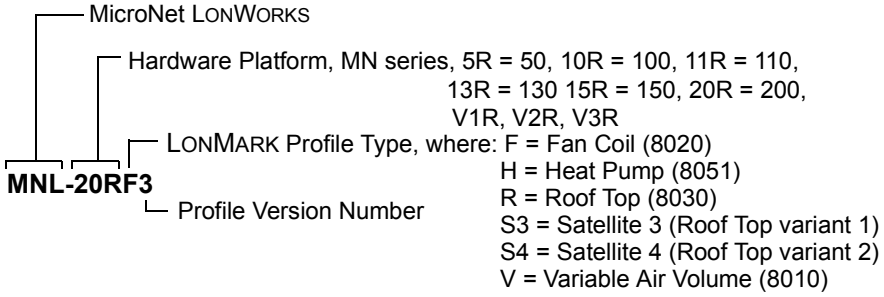
Each LONMARK MicroNet controller (MN 50, 100, 110, 130, 150, 200, and VAV series) has one profile in Read Only Memory (ROM). Each profile contains all the mandatory and most of the optional network variables and configuration properties defined by the LONMARK Association. Each also has additional Invensys-defined profile items in the “manufacturer defined section”. Profile items are generally described (typical usage) in following sections.

Note: Throughout the various Invensys LONMARK profiles, general-purpose type network variables are given a “Satellite” description, such as Satellite Temperature Input 1 (nviSatTemp1). The actual usage definition of any item depends on the specific application in the controller. Not surprisingly, the two “Satellite profiles”, Satellite 1 and Satellite 2, have the most “Satellite” or variable items. All four Satellite profiles are certified by LONMARK using all Roof Top Unit mandatory profile items. They also include more general-purpose items in the “manufacturer-defined” section to provide additional flexibility for engineering custom applications.

Profile by Model Number

Model Number Code

MicroNet standard controllers have model numbers coded with the following information:.


MicroNet LONWORKS
Hardware Platform, MN series, 5R = 50, 10R = 100, 11R = 110,
13R = 130 15R = 150, 20R = 200,
V1R, V2R, V3R
LONMARK Profile Type, where: F = Fan Coil (8020)
H = Heat Pump (8051)
R = Roof Top (8030)
S3 = Satellite 3 (Roof Top variant 1)
S4 = Satellite 4 (Roof Top variant 2)
V = Variable Air Volume (8010)
MNL-20RF3
Profile Version Number

- All VAV series controllers (V1R, V2R, and V3R) are available only with a VAV controller profile.
- The MN 50, 100, 150, and 200 controller platforms are available in any of the other profiles: Fan Coil, Heat Pump, Roof Top, Satellite 3, or Satellite 4.
- The MN 110 and MN 130 controller platforms are available in Fan Coil profile only.
- Profile version number “by model number” does not apply to controller models with Satellite profiles. Satellite 1, 2, 3, and 4 all have version 1 profiles.

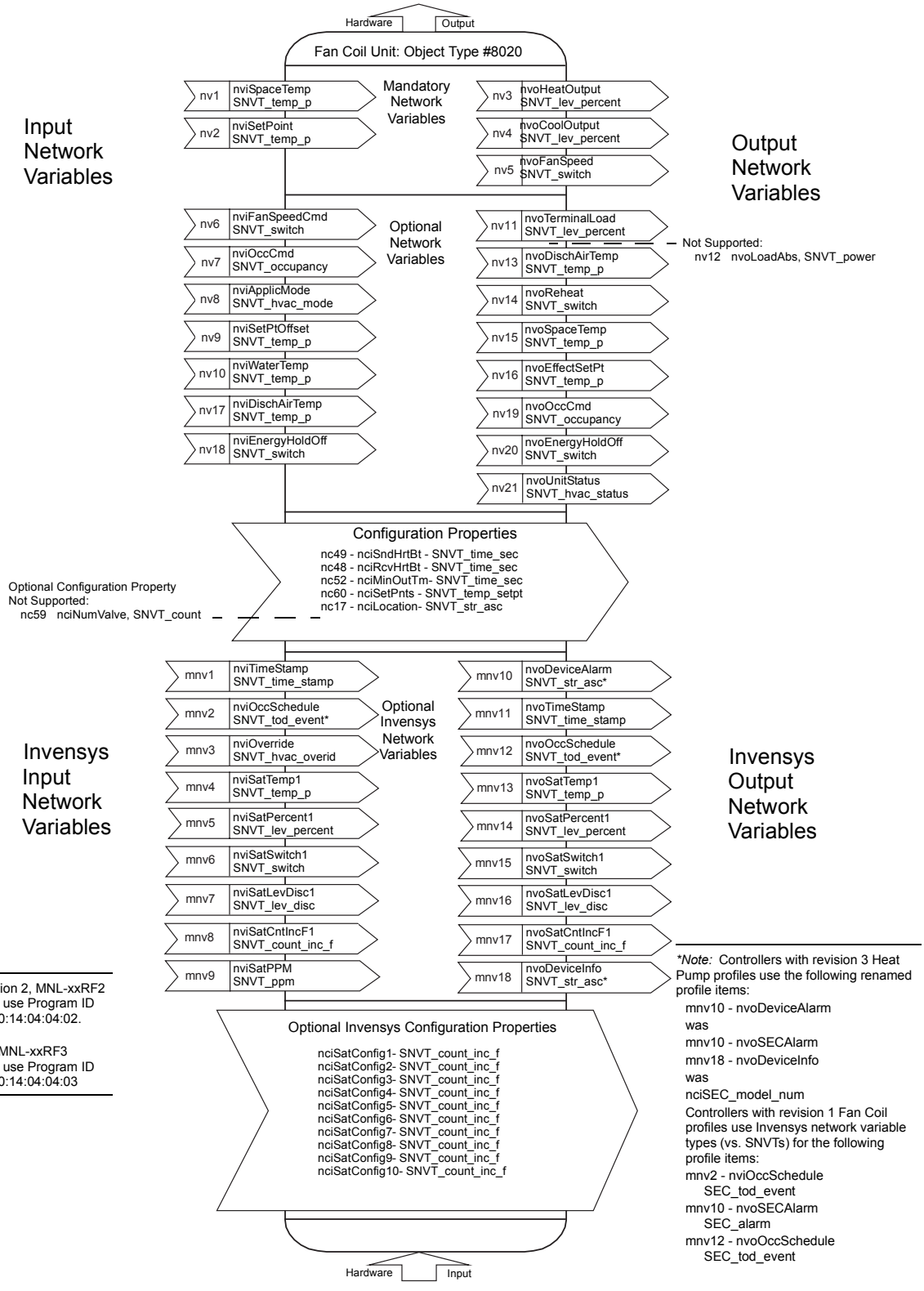
Profile Versions

MN 100, 200, and VAV series controllers were originally released with version 1 profiles, but currently are released with version 3 profiles. Differences between profile versions are mostly minor, and are noted in each profile in this chapter. See also “Fan Coil Profile”, next.

Note: Controllers with Satellite profile 3 or 4 also have the latest changes incorporated in the other version 3 profiles.

Fan Coil Profile

The Invensys Fan Coil profile is graphically shown below Figure-1.



Figure–B.1 Fan Coil Profile

Fan Coil Profile Quick Reference

For each Fan Coil profile item, Table–B.1 includes the network variable (NV) index number, self-doc index number, item name, general description, SNVT type, element names and numbers as applicable, and point type (AV = Analog Value, DV = Digital Value). An example Profile tag shape is shown in the header above each class of profile items.

Table–B.1 Invensys LONMARK Fan Coil Profile Items: Version 2 - MNL-xxRF2 Program ID 80:00:16:50:14:04:04:02 and Version 3 - MNL-xxRF3 Program ID 80:00:16:50:14:04:04:03.

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Type	SNVT Num.	Elements	Element Numbers	Point Type
	nvi*Name*	**Mandatory Network Variable Inputs**						
0	nv1	nviSpaceTemp	Space Temperature (In)	SNVT_temp_p	105	—	—	AV
1	nv2	nviSetPoint	Single Setpoint	SNVT_temp_p	105	—	—	AV
	nvo*Name*	**Mandatory Network Variable Outputs**						
2	nv3	nvoHeatOutput	Current Position/Percentage of Heat Output	SNVT_lev_percent	81	—	—	AV
3	nv4	nvoCoolOutput	Current Position of Cool Actuator	SNVT_lev_percent	81	—	—	AV
4	nv5	nvoFanSpeed	Current Fan Speed (Out)	SNVT_switch	95	.Value .State	1 2	AV DV
	nvi*Name*	**Optional Network Variable Inputs**						
5	nv6	nviFanSpeedCmd	Fan Speed Command (In)	SNVT_switch	95	.Value .State	1 2	AV DV
6	nv7	nviOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
7	nv8	nviApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
8	nv9	nviSetPtOffset	Single Setpoint Offset	SNVT_temp_p	105	—	—	AV
9	nv10	nviWaterTemp	Water Temperature (In) for Providing Automatic Heat/Cool Switchover.	SNVT_temp_p	105	—	—	AV
10	nv17	nviDischAirTemp	Discharge Air Temperature (In)	SNVT_temp_p	105	—	—	AV
11	nv18	nviEnergyHoldOff	Energy Hold Off Command	SNVT_switch	95	.Value .State	1 2	AV DV
	nvo*Name*	**Optional Network Variable Outputs**						
12	nv11	nvoTerminalLoad	Current Heat/Cool Energy Demand of Fan Coil Controller	SNVT_lev_percent	81	—	—	AV
—	nv121	nvoLoadAbs1	Absolute Power Consumption1	SNVT_power1	27	1Note: Not Supported		
13	nv13	nvoDischAirTemp	Discharge Air Temperature (Out)	SNVT_temp_p	105	—	—	AV
14	nv14	nvoReheat	Current State of a Multi-Stage Reheat Output	SNVT_switch	95	.Value .State	1 2	AV DV
15	nv15	nvoSpaceTemp	Space Temperature (Out)	SNVT_temp_p	105	—	—	AV
16	nv16	nvoEffectSetPt	Setpoint Value Used	SNVT_temp_p	105	—	—	AV
17	nv19	nvoOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
18	nv20	nvoEnergyHoldOff	State of Energy Hold-off Device Hardwired to Controller	SNVT_switch	95	.Value .State	1 2	AV DV
19	nv21	nvoUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode .Heat2 .Heat1 .Cool .Econ .Fan .Alarm	1 2 3 4 5 6 7	AV AV AV AV AV AV AV

Table–B.1 Invensys LONMARK Fan Coil Profile Items: Version 2 - MNL-xxRF2 Program ID 80:00:16:50:14:04:04:02 and Version 3 - MNL-xxRF3 Program ID 80:00:16:50:14:04:04:03. (Continued)

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Type	SNVT Num.	Elements	Element Numbers	Point Type
	nci*Name*	**Configuration Properties**						
37	nc60	nciSetPnts	Default Setpoints	SNVT_temp_setpt	106	.OccCool .SBCool .UnoccCool .OccHeat .SBHeat .UnoccHeat	1 2 3 4 5 6	AV AV AV AV AV AV
38	nc49	nciSndHrtBt	Send Heartbeat (see page 547)	SNVT_time_sec	107	—	—	AV
39	nc52	nciMinOutTm	Minimum Time between Output Transmissions (see page 547)	SNVT_time_sec	107	—	—	AV
40	nc48	nciRcvHrtBt	Receive Heartbeat (see page 545)	SNVT_time_sec	107	—	—	AV
41	nc17	nciLocation	Text Descriptor for Physical Location	SNVT_str_asc	36	1Not Supported		
—	nc591	nciNumValve1	Number of Valves1	SNVT_count1	8	1Not Supported		
	nvi*Name*	**Invensys Network Variable Inputs**						
20	mnv1	nviTimeStamp	Time Input from External Device *Note:* Rev.2 or higher profile uses this NVI to sync the controller clock, No usable NVI tags in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
21	mnv2	nviOccSchedule	Occupancy Scheduling Input	SNVT_tod_event (SEC_tod_event if revision 1 profile)	128	.Current .Next .Time	1 2 3	AV AV AV
22	mnv3	nviOverride	Override Input	SNVT_hvac_overid	111	.State .Percent .Flow	1 2 3	AV AV AV
23	mnv4	nviSatTemp1	Satellite Temperature Input 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
24	mnv5	nviSatPercent1	Satellite Percent Input 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
25	mnv6	nviSatSwitch1	Satellite Switch Input 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
26	mnv7	nviSatLevDisc1	Satellite Level Discrete Input 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
27	mnv8	nviSatCntIncF1	Satellite Count Incremental Input (defined by the application)	SNVT_count_inc_f	52	—	—	AV
28	mnv9	nviSatPPM	Satellite Parts Per Million Input (defined by the application)	SNVT_ppm	29	—	—	AV

**Table–B.1 Invensys LONMARK Fan Coil Profile Items:
Version 2 - MNL-xxRF2 Program ID 80:00:16:50:14:04:04:02 and
Version 3 - MNL-xxRF3 Program ID 80:00:16:50:14:04:04:03. (Continued)**

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Type	SNVT Num.	Elements	Element Numbers	Point Type
	nvo*Name*	**Invensys Network Variable Outputs**						
29	mnv10	nvoDeviceAlarm (nvoSECAlarm if version 1 or 2 profile)	Alarm Output (Future Use, No WP Tech visibility)	SNVT_str_asc (SEC_alarm if version 1 profile)	36	1Not Supported		
30	mnv11	nvoTimeStamp	Time Output *Note:* Rev.2 or higher profile uses this NVO to transmit the time and date of the controller clock. No usable NVO tags in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
31	mnv12	nvoOccSchedule	Occupancy Scheduling Output	SNVT_tod_event (SEC_tod_event if revision 1 profile)	128	.Current .Next .Time	1 2 3	AV AV AV
32	mnv13	nvoSatTemp1	Satellite Temperature Output 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
33	mnv14	nvoSatPercent1	Satellite Percent Output 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
34	mnv15	nvoSatSwitch1	Satellite Switch Output 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
35	mnv16	nvoSatLevDisc1	Satellite Level Discrete Output 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
36	mnv17	nvoSatCntIncF1	Satellite Count Incremental Output 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
52[a]	mnv18	nvoDeviceInfo	Controller Model Number, Firmware Version, Diagnostic Status, and Application Name.	SNVT_str_asc	36	1Not Supported		
	nci*Name*	**Invensys Configuration Properties**						
42	mnc1	nciSatConfig1	Satellite Configuration Property 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
43	mnc3	nciSatConfig2	Satellite Configuration Property 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
44	mnc4	nciSatConfig3	Satellite Configuration Property 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
45	mnc5	nciSatConfig4	Satellite Configuration Property 4 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
46	mnc6	nciSatConfig5	Satellite Configuration Property 5 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
47	mnc7	nciSatConfig6	Satellite Configuration Property 6 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
48	mnc8	nciSatConfig7	Satellite Configuration Property 7 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
49	mnc9	nciSatConfig8	Satellite Configuration Property 8 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
50	mnc10	nciSatConfig9	Satellite Configuration Property 9 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

**Table–B.1 Invensys LONMARK Fan Coil Profile Items:
Version 2 - MNL-xxRF2 Program ID 80:00:16:50:14:04:04:02 and
Version 3 - MNL-xxRF3 Program ID 80:00:16:50:14:04:04:03. (Continued)**

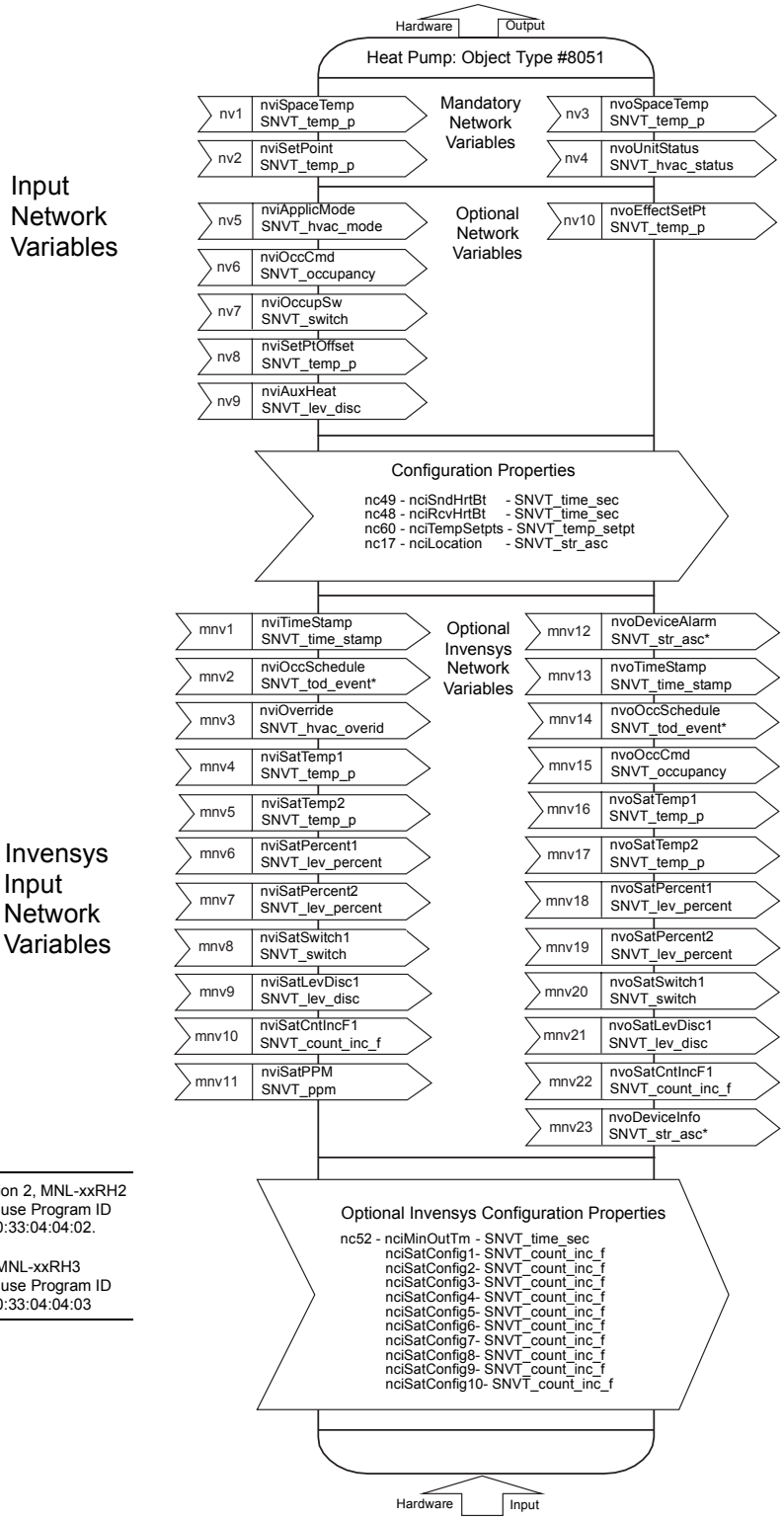
NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Type	SNVT Num.	Elements	Element Numbers	Point Type
51	mnc11	nciSatConfig10	Satellite Configuration Property 10 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
52[b]	mnc2[b]	nciSECModelNum	Controller Part Number and Application Number	SNVT_str_asc	36	1Not Supported		

a. In version 3 profile only

b. In version 1 or 2 profile only

Heat Pump Profile

The Invensys Heat Pump profile is graphically shown below Figure-2.



Note: Version 2, MNL-xxRH2 Controllers use Program ID 80:00:16:50:33:04:04:02.

Version 3, MNL-xxRH3 Controllers use Program ID 80:00:16:50:33:04:04:03

Note: Controllers with revision 3 Heat Pump profiles use the following renamed profile items:

mnv12 - nvoDeviceAlarm was mnv12 - nvoSECAlarm

mnv23 - nvoDeviceInfo was nciSEC_model_num

Controllers with revision 1 Heat Pump profiles use Invensys network variable types (vs. SNVTs) for the following profile items:

mnv2 - nviOccSchedule SEC_tod_event

mnv12 - nvoSECAlarm SEC_alarm

mnv14 - nvoOccSchedule SEC_tod_event

However, data structures are equivalent.

Figure–B.2 Heat Pump Profile

Heat Pump Profile Quick Reference

For each Heat Pump profile item, Table–B.2 includes the network variable (NV) index number, self-doc index number, item name, general description, SNVT type, element names and numbers as applicable, and point type (AV = Analog Value, DV = Digital Value). An example Profile tag shape is shown in the header above each class of profile items.

Table–B.2 Invensys LONMARK Heat Pump Profile Items: Version 2 - Program ID 80:00:16:50:33:04:04:02 and Version 3 - Program ID 80:00:16:50:33:04:04:03

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nviName	**Mandatory Network Variable Inputs**						
0	nv1	nviSpaceTemp	Space Temperature (In)	SNVT_temp_p	105	—	—	AV
1	nv2	nviSetPoint	Single Setpoint	SNVT_temp_p	105	—	—	AV
	nvoName	**Mandatory Network Variable Outputs**						
2	nv3	nvoSpaceTemp	Space Temperature (Out)	SNVT_temp_p	105	—	—	AV
3	nv4	nvoUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode	1	AV
						.Heat2	2	AV
						.Heat1	3	AV
						.Cool	4	AV
						.Econ	5	AV
						.Fan	6	AV
						.Alarm	7	AV
	nviName	**Optional Network Variable Inputs**						
4	nv5	nviApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
5	nv6	nviOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
6	nv7	nviOccupSw	Occupancy Status (2-State)	SNVT_switch	95	.Value	1	AV
						.State	2	DV
7	nv8	nviSetPtOffset	Single Setpoint Offset	SNVT_temp_p	105	—	—	AV
8	nv9	nviAuxHeat	Auxiliary Heat Enable	SNVT_lev_disc	22	—	—	AV
	nvoName	**Optional Network Variable Outputs**						
9	nv10	nvoEffectSetPt	Setpoint Value Used	SNVT_temp_p	105	—	—	AV
	nciName	**Configuration Properties**						
32	nc60	nciTempSetPts	Default Setpoints	SNVT_temp_setpt	106	.OccCool	1	AV
						.SBCool	2	AV
						.UnoccCool	3	AV
						.OccHeat	4	AV
						.SBHeat	5	AV
						.UnoccHeat	6	AV
33	nc49	nciSndHrtBt	Send Heartbeat (see page 547)	SNVT_time_sec	107	—	—	AV
34	nc48	nciRcvHrtBt	Receive Heartbeat (see page 545)	SNVT_time_sec	107	—	—	AV
35	nc17	nciLocation	Text Descriptor for Physical Location	SNVT_str_asc	36	1Not Supported		

Table–B.2 Invensys LONMARK Heat Pump Profile Items: Version 2 - Program ID 80:00:16:50:33:04:04:02 and Version 3 - Program ID 80:00:16:50:33:04:04:03

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs**						
10	mnv1	nviTimeStamp	Time Input from External Device *Note:* Rev.2 or higher profile uses this NVI to sync the controller clock; no usable NVI tags are in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
11	mnv2	nviOccSchedule	Occupancy Scheduling Input	SNVT_tod_event (SEC_tod_event if revision 1 profile)	128	.Current .Next .Time	1 2 3	AV AV AV
12	mnv3	nviOverride	Override Input	SNVT_hvac_overid	111	.State .Percent .Flow	1 2 3	AV AV AV
13	mnv4	nviSatTemp1	Satellite Temperature Input 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
14	mnv5	nviSatTemp2	Satellite Temperature Input 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
15	mnv6	nviSatPercent1	Satellite Percent Input 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
16	mnv7	nviSatPercent2	Satellite Percent Input 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
17	mnv8	nviSatSwitch1	Satellite Switch Input 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
18	mnv9	nviSatLevDisc1	Satellite Level Discrete Input 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
19	mnv10	nviSatCntIncF1	Satellite Count Incremental Input 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
20	mnv11	nviSatPPM	Satellite Parts Per Million Input (defined by the application)	SNVT_ppm	29	—	—	AV
	nvo*Name*	**Invensys Network Variable Outputs**						
21	mnv12	nvoDeviceAlarm (nvoSECAlarm in version 1 or 2 profile)	Alarm Output (Future Use, No WP Tech visibility)	SNVT_str_asc (SEC_alarm if revision 1 profile)	36	1Not Supported		
22	mnv13	nvoTimeStamp	Time Output *Note:* Rev.2 or higher profile uses this NVO to transmit the time and date of the controller clock. No usable NVO tags in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
23	mnv14	nvoOccSchedule	Occupancy Scheduling Output	SNVT_tod_event (SEC_tod_event if revision 1 profile)	128	.Current .Next .Time	1 2 3	AV AV AV
24	mnv15	nvoOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV

Table–B.2 Invensys LONMARK Heat Pump Profile Items: Version 2 - Program ID 80:00:16:50:33:04:04:02 and Version 3 - Program ID 80:00:16:50:33:04:04:03

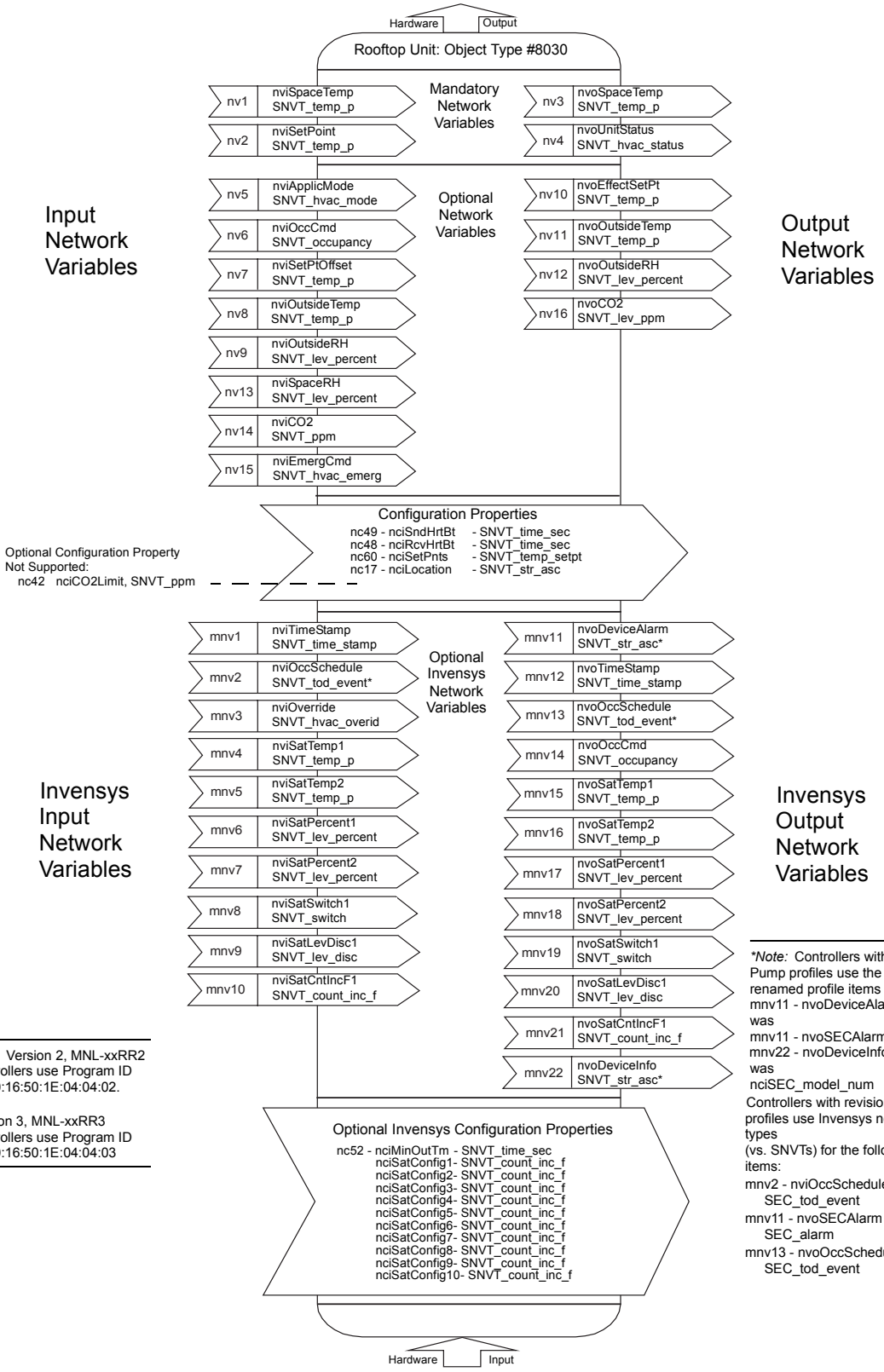
NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvo*Name*	**Invensys Network Variable Outputs (continued)**						
25	mnv16	nvoSatTemp1	Satellite Temperature Output 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
26	mnv17	nvoSatTemp2	Satellite Temperature Output 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
27	mnv18	nvoSatPercent1	Satellite Percent Output 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
28	mnv19	nvoSatPercent2	Satellite Percent Output 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
29	mnv20	nvoSatSwitch1	Satellite Switch Output 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
30	mnv21	nvoSatLevDisc1	Satellite Level Discrete Output 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
31	mnv22	nvoSatCntIncF1	Satellite Count Incremental Output 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
47[a]	mnv23	nvoDeviceInfo	Controller Model Number, Firmware Version, Diagnostic Status, and Application Name	SNVT_str_asc	36	1Not Supported		
	nci*Name*	**Invensys Configuration Properties**						
36	nc52	nciMinOutTm	Minimum Time between Output Transmissions (see page 547)	SNVT_time_sec	107	—	—	AV
37	mnc1	nciSatConfig1	Satellite Configuration Property 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
38	mnc3	nciSatConfig2	Satellite Configuration Property 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
39	mnc4	nciSatConfig3	Satellite Configuration Property 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
40	mnc5	nciSatConfig4	Satellite Configuration Property 4 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
41	mnc6	nciSatConfig5	Satellite Configuration Property 5 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
42	mnc7	nciSatConfig6	Satellite Configuration Property 6 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
43	mnc8	nciSatConfig7	Satellite Configuration Property 7 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
44	mnc9	nciSatConfig8	Satellite Configuration Property 8 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
45	mnc10	nciSatConfig9	Satellite Configuration Property 9 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
46	mnc11	nciSatConfig10	Satellite Configuration Property 10 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
47[b]	mnc2[b]	nciSECModelNum	Controller Part Number and Application Number	SNVT_str_asc	36	1Not Supported		

a. In version 3 profile only

b. In version 1 or 2 profile only

Roof Top Unit Profile

The Invensys Roof Top Unit profile is graphically shown below Figure-3.



Figure–B.3 Roof Top Unit Profile

Roof Top Unit Profile Quick Reference

For each Roof Top Unit profile item, Table–B.3 includes the network variable (NV) index number, self-doc index number, item name, general description, SNVT type, element names and numbers as applicable, and point type (AV = Analog Value, DV = Digital Value). An example Profile tag shape is shown in the header above each class of profile items.

Table–B.3 Invensys LONMARK Roof Top Unit Profile Items: Version 2 - Program ID 80:00:16:50:1E:04:04:02 and Version 3- Program ID 80:00:16:50:1E:04:04:03

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nviName	**Mandatory Network Variable Inputs**						
0	nv1	nviSpaceTemp	Space Temperature (In)	SNVT_temp_p	105	—	—	AV
1	nv2	nviSetPoint	Single Setpoint	SNVT_temp_p	105	—	—	AV
	nvoName	**Mandatory Network Variable Outputs**						
2	nv3	nvoSpaceTemp	Space Temperature (Out)	SNVT_temp_p	105	—	—	AV
3	nv4	nvoUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode	1	AV
						.Heat2	2	AV
						.Heat1	3	AV
						.Cool	4	AV
						.Econ	5	AV
						.Fan	6	AV
						.Alarm	7	AV
	nviName	**Optional Network Variable Inputs**						
4	nv5	nviApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
5	nv6	nviOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
6	nv7	nviSetPtOffset	Single Setpoint Offset	SNVT_temp_p	105	—	—	AV
7	nv8	nviOutsideTemp	Outside Air Temperature (In)	SNVT_temp_p	105	—	—	AV
8	nv9	nviOutsideRH	Outside Air Relative Humidity (In)	SNVT_lev_percent	81	—	—	AV
9	nv13	nviSpaceRH	Space Relative Humidity (In)	SNVT_lev_percent	81	—	—	AV
10	nv14	nviCO2	CO2 Sensor in PPM (In)	SNVT_ppm	29	—	—	AV
11	nv15	nviEmergCmd	Emergency Command (In)	SNVT_hvac_emerg	103	—	—	AV
	nvoName	**Optional Network Variable Outputs**						
12	nv10	nvoEffectSetPt	Setpoint Value Used	SNVT_temp_p	105	—	—	AV
13	nv11	nvoOutsideTemp	Outside Air Temperature (Out)	SNVT_temp_p	105	—	—	AV
14	nv12	nvoOutsideRH	Outside Air Relative Humidity (Out)	SNVT_lev_percent	81	—	—	AV
15	nv16	nvoCO2	CO2 Sensor in PPM (Out)	SNVT_ppm	29	—	—	AV
	nciName	**Configuration Properties**						
37	nc60	nciSetPnts	Default Setpoints	SNVT_temp_setpt	106	.OccCool	1	AV
						.SBCool	2	AV
						.UnoccCool	3	AV
						.OccHeat	4	AV
						.SBHeat	5	AV
						.UnoccHeat	6	AV
38	nc49	nciSndHrtBt	Send Heartbeat (see page 547)	SNVT_time_sec	107	—	—	AV
39	nc48	nciRcvHrtBt	Receive Heartbeat (see page 545)	SNVT_time_sec	107	—	—	AV
40	nc17	nciLocation	Text Descriptor for Physical Location	SNVT_str_asc	36	1Not Supported		
—	nc421	nciCO2Limit1	CO2 Level Limit1	SNVT_ppm	29	1Not Supported		

Table–B.3 Invensys LONMARK Roof Top Unit Profile Items: Version 2 - Program ID 80:00:16:50:1E:04:04:02 (Continued)and Version 3- Program ID 80:00:16:50:1E:04:04:03

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs**						
16	mnv1	nviTimeStamp	Time Input from External Device *Note:* Rev.2 or higher profile uses this NVI to sync the controller clock, No usable NVI tags in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
17	mnv2	nviOccSchedule	Occupancy Scheduling Input	SNVT_tod_event (SEC_tod_event if revision 1 profile)	128	.Current .Next .Time	1 2 3	AV AV AV
18	mnv3	nviOverride	Override Input	SNVT_hvac_overid	111	.State .Percent .Flow	1 2 3	AV AV AV
19	mnv4	nviSatTemp1	Satellite Temperature Input 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
20	mnv5	nviSatTemp2	Satellite Temperature Input 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
21	mnv6	nviSatPercent1	Satellite Percent Input 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
22	mnv7	nviSatPercent2	Satellite Percent Input 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
23	mnv8	nviSatSwitch1	Satellite Switch Input 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
24	mnv9	nviSatLevDisc1	Satellite Level Discrete Input 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
25	mnv10	nviSatCntIncF1	Satellite Count Incremental Input 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
	nvo*Name*	**Invensys Network Variable Outputs**						
26	mnv11	nvoDeviceAlarm (nvoSECAlarm if version 1 or 2 profile)	Alarm Output (Future Use, No WP Tech visibility)	SNVT_str_asc (SEC_alarm if revision 1 profile)	36	1Not Supported		
27	mnv12	nvoTimeStamp	Time Output *Note:* Rev.2 or higher profile uses this NVO to transmit the time and date of the controller clock. No usable NVO tags in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
28	mnv13	nvoOccSchedule	Occupancy Scheduling Output	SNVT_tod_event (SEC_tod_event if revision 1 profile)	128	.Current .Next .Time	1 2 3	AV AV AV
29	mnv14	nvoOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
30	mnv15	nvoSatTemp1	Satellite Temperature Output 1 (defined by the application)	SNVT_temp_p	105	—	—	AV

Table–B.3 Invensys LONMARK Roof Top Unit Profile Items: Version 2 - Program ID 80:00:16:50:1E:04:04:02 (Continued)and Version 3- Program ID 80:00:16:50:1E:04:04:03

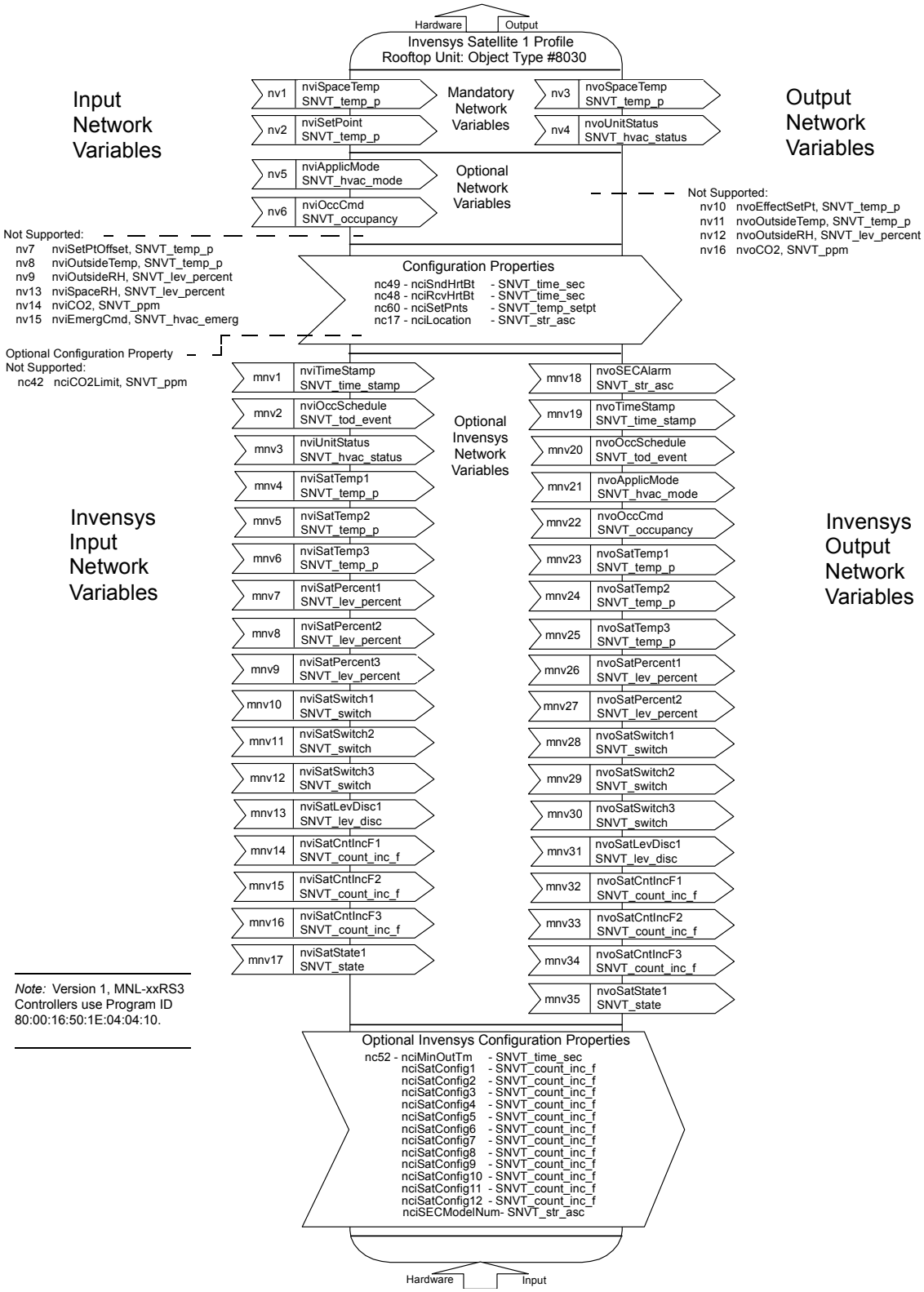
NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvo*Name*	**Invensys Network Variable Outputs (continued)**						
31	mnv16	nvoSatTemp2	Satellite Temperature Output 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
32	mnv17	nvoSatPercent1	Satellite Percent Output 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
33	mnv18	nvoSatPercent2	Satellite Percent Output 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
34	mnv19	nvoSatSwitch1	Satellite Switch Output 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
35	mnv20	nvoSatLevDisc1	Satellite Level Discrete Output 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
36	mnv21	nvoSatCntIncF1	Satellite Count Incremental Output 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
52[a]	mnv22	nvoDeviceInfo	Controller Model Number, Firmware Version, Diagnostic Status, and Application name	SNVT_str_asc	36	1Not Supported		
	nci*Name*	**Invensys Configuration Properties**						
41	nc52	nciMinOutTm	Minimum Time between Output Transmissions (see page 547)	SNVT_time_sec	107	—	—	AV
42	mnc1	nciSatConfig1	Satellite Configuration Property 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
43	mnc3	nciSatConfig2	Satellite Configuration Property 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
44	mnc4	nciSatConfig3	Satellite Configuration Property 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
45	mnc5	nciSatConfig4	Satellite Configuration Property 4 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
46	mnc6	nciSatConfig5	Satellite Configuration Property 5 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
47	mnc7	nciSatConfig6	Satellite Configuration Property 6 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
48	mnc8	nciSatConfig7	Satellite Configuration Property 7 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
49	mnc9	nciSatConfig8	Satellite Configuration Property 8 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
50	mnc10	nciSatConfig9	Satellite Configuration Property 9 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
51	mnc11	nciSatConfig10	Satellite Configuration Property 10 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
52[b]	mnc2[b]	nciSECModelNum	Controller Part Number and Application Number	SNVT_str_asc	36	1Not Supported		

a. In version 3 profile only

b. In version 1 or 2 profile only

Satellite 1 Profile

The Invensys Satellite 1 profile is graphically shown below Figure-3.



Figure–B.4 Satellite 1 Profile

Satellite 1 Profile Quick Reference

For each Satellite 1 profile item, Table–B.6 includes the network variable (NV) index number, self-doc index number, item name, general description, SNVT type, element names and numbers as applicable, and point type (AV = Analog Value, DV = Digital Value). An example Profile tag shape is shown in the header above each class of profile items.

Table–B.4 Invensys Satellite 1 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:10.

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nviName	**Mandatory Network Variable Inputs**						
0	nv1	nviSpaceTemp	Space Temperature (In)	SNVT_temp_p	105	—	—	AV
1	nv2	nviSetPoint	Single Setpoint	SNVT_temp_p	105	—	—	AV
	nvoName	**Mandatory Network Variable Outputs**						
2	nv3	nvoSpaceTemp	Space Temperature (Out)	SNVT_temp_p	105	—	—	AV
3	nv4	nvoUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode	1	AV
						.Heat2	2	AV
						.Heat1	3	AV
						.Cool	4	AV
						.Econ	5	AV
						.Fan	6	AV
						.Alarm	7	AV
	nviName	**Optional Network Variable Inputs**						
4	nv5	nviApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
5	nv6	nviOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
—	nv71	nviSetPtOffset1	Single Setpoint Offset1	SNVT_temp_p	105	1Not Supported		
—	nv81	nviOutsideTemp1	Outside Air Temperature (In)1	SNVT_temp_p	105	1Not Supported		
—	nv91	nviOutsideRH1	Outside Air Relative Humidity (In)1	SNVT_lev_percent	81	1Not Supported		
—	nv131	nviSpaceRH1	Space Relative Humidity (In)1	SNVT_lev_percent	81	1Not Supported		
—	nv141	nviCO21	CO2 Sensor in PPM (In)1	SNVT_ppm	29	1Not Supported		
—	nv151	nviEmergCmd1	Emergency Command (In)1	SNVT_hvac_emerg	103	1Not Supported		
	nvoName	**Optional Network Variable Outputs**						
—	nv101	nvoEffectSetPt1	Setpoint Value Used1	SNVT_temp_p	105	1Not Supported		
—	nv111	nvoOutsideTemp1	Outside Air Temperature (Out)1	SNVT_temp_p	105	1Not Supported		
—	nv121	nvoOutsideRH1	Outside Air Relative Humidity (Out)1	SNVT_lev_percent	81	1Not Supported		
—	nv161	nvoCO21	CO2 Sensor in PPM (Out)1	SNVT_ppm	29	1Not Supported		
	nciName	**Configuration Properties**						
41	nc60	nciSetPnts	Default Setpoints	SNVT_temp_setpt	106	.OccCool	1	AV
						.SBCool	2	AV
						.UnoccCool	3	AV
						.OccHeat	4	AV
						.SBHeat	5	AV
						.UnoccHeat	6	AV
42	nc49	nciSndHrtBt	Send Heartbeat (see page 547)	SNVT_time_sec	107	—	—	AV
43	nc48	nciRcvHrtBt	Receive Heartbeat (see page 545)	SNVT_time_sec	107	—	—	AV
44	nc17	nciLocation	Text Descriptor for Physical Location	SNVT_str_asc	36	1Not Supported		
—	nc421	nciCO2Limit1	CO2 Level Limit1	SNVT_ppm	29	1Not Supported		

Table–B.4 Invensys Satellite 1 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:10. (Continued)

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs**						
6	mnv1	nviTimeStamp	Time Input from External Device *Note:* This NVI is used to sync the controller clock; no usable NVI tags are in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
7	mnv2	nviOccSchedule	Occupancy Scheduling Input	SNVT_tod_event	128	.Current .Next .Time	1 2 3	AV AV AV
8	mnv3	nviUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode .Heat2 .Heat1 .Cool .Econ .Fan .Alarm	1 2 3 4 5 6 7	AV AV AV AV AV AV AV
9	mnv4	nviSatTemp1	Satellite Temperature Input 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
10	mnv5	nviSatTemp2	Satellite Temperature Input 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
11	mnv6	nviSatTemp3	Satellite Temperature Input 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
12	mnv7	nviSatPercent1	Satellite Percent Input 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
13	mnv8	nviSatPercent2	Satellite Percent Input 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
14	mnv9	nviSatPercent3	Satellite Percent Input 3 (defined by the application)	SNVT_lev_percent	81	—	—	AV
15	mnv10	nviSatSwitch1	Satellite Switch Input 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
16	mnv11	nviSatSwitch2	Satellite Switch Input 2 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
17	mnv12	nviSatSwitch3	Satellite Switch Input 3 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
18	mnv13	nviSatLevDisc1	Satellite Level Discrete Input 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
19	mnv14	nviSatCntIncF1	Satellite Count Incremental Input 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
20	mnv15	nviSatCntIncF2	Satellite Count Incremental Input 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
21	mnv16	nviSatCntIncF3	Satellite Count Incremental Input 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

Table–B.4 Invensys Satellite 1 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:10. (Continued)

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs (continued)**						
22	mnv17	nviSatState1	Satellite State Input 1 (defined by the application)	SNVT_state	83	.State0	1	DV
						.State1	2	DV
						.State2	3	DV
						.State3	4	DV
						.State4	5	DV
						.State5	6	DV
						.State6	7	DV
						.State7	8	DV
						.State8	9	DV
						.State9	10	DV
						.State10	11	DV
						.State11	12	DV
						.State12	13	DV
						.State13	14	DV
						.State14	15	DV
						.State15	16	DV
	nvo*Name*	**Invensys Network Variable Outputs**						
23	mnv18	nvoSECAlarm	Alarm Output (Future Use, No WP Tech visibility)	SNVT_str_asc (SEC_alarm if revision 1 profile)	36	1Not Supported		
24	mnv19	nvoTimeStamp	Time Output *Note:* This NVO transmits the time and date of the controller clock. No usable NVO tags are in WP Tech.	SNVT_time_stamp	84	(See *Note*)		
						.Year	1	AV
						.Month	2	AV
						.Day	3	AV
						.Hour	4	AV
						.Minute	5	AV
						.Second	6	AV
25	mnv20	nvoOccSchedule	Occupancy Scheduling Output	SNVT_tod_event	128	.Current	1	AV
						.Next	2	AV
						.Time	3	AV
26	mnv21	nvoApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
27	mnv22	nvoOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
28	mnv23	nvoSatTemp1	Satellite Temperature Output 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
29	mnv24	nvoSatTemp2	Satellite Temperature Output 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
30	mnv25	nvoSatTemp3	Satellite Temperature Output 3 (defined by the application)	SNVT_temp_p	105	—	—	AV
31	mnv26	nvoSatPercent1	Satellite Percent Output 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
32	mnv27	nvoSatPercent2	Satellite Percent Output 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
33	mnv28	nvoSatSwitch1	Satellite Switch Output 1 (defined by the application)	SNVT_switch	95	.Value	1	AV
						.State	2	DV
34	mnv29	nvoSatSwitch2	Satellite Switch Output 2 (defined by the application)	SNVT_switch	95	.Value	1	AV
						.State	2	DV

Table–B.4 Invensys Satellite 1 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:10. (Continued)

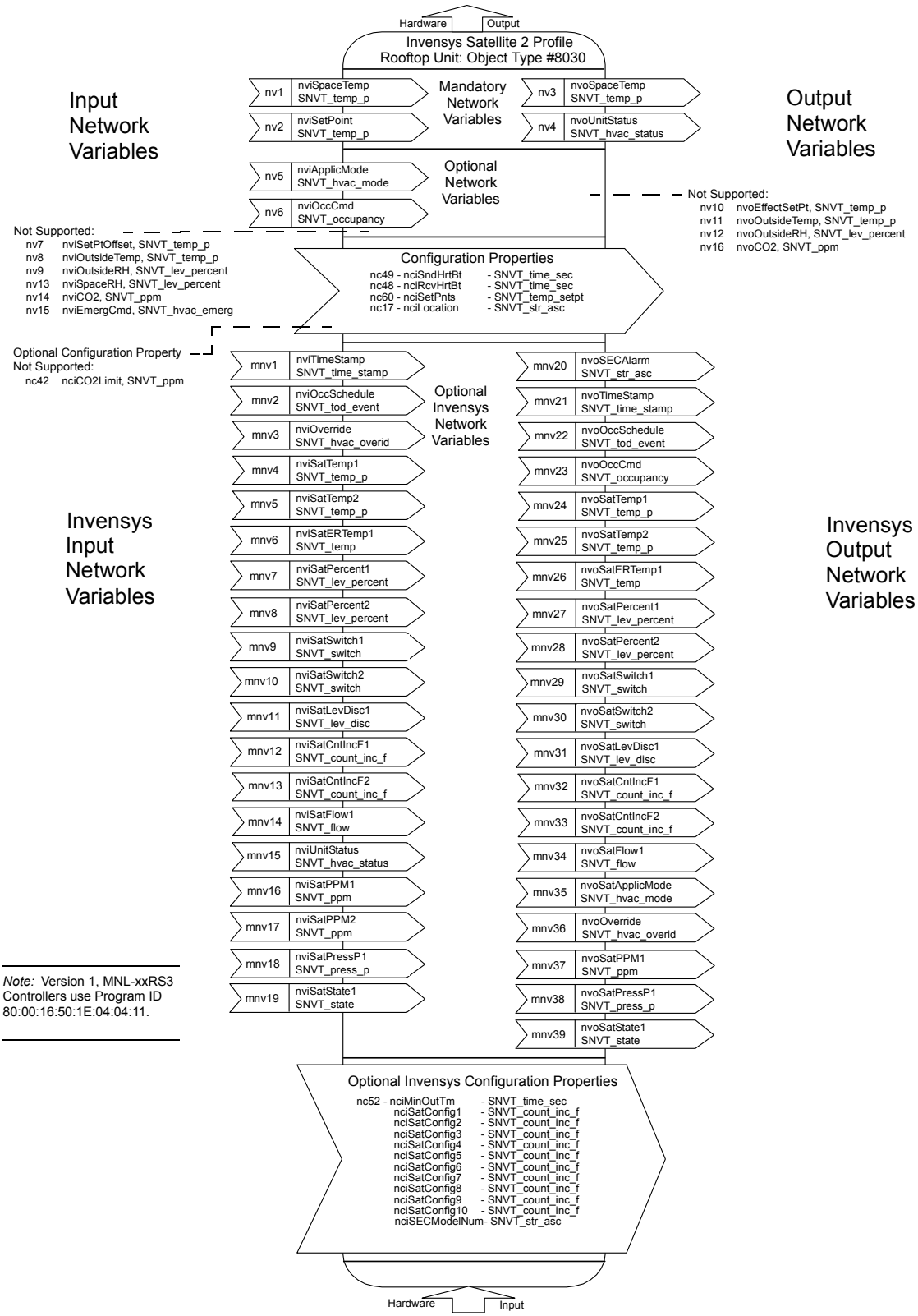
NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvo*Name*	**Invensys Network Variable Outputs (continued)**						
35	mnv30	nvoSatSwitch3	Satellite Switch Output 3 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
36	mnv31	nvoSatLevDisc1	Satellite Level Discrete Output 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
37	mnv32	nvoSatCntIncF1	Satellite Count Incremental Output 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
38	mnv33	nvoSatCntIncF2	Satellite Count Incremental Output 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
39	mnv34	nvoSatCntIncF3	Satellite Count Incremental Output 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
40	mnv35	nvoSatState1	Satellite State Output 1 (defined by the application)	SNVT_state	83	.State0 .State1 .State2 .State3 .State4 .State5 .State6 .State7 .State8 .State9 .State10 .State11 .State12 .State13 .State14 .State15	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	DV DV DV DV DV DV DV DV DV DV DV DV DV DV DV DV
	nci*Name*	**Invensys Configuration Properties**						
45	nc52	nciMinOutTm	Minimum Time between Output Transmissions (see page 547)	SNVT_time_sec	107	—	—	AV
46	mnc1	nciSatConfig1	Satellite Configuration Property 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
47	mnc3	nciSatConfig2	Satellite Configuration Property 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
48	mnc4	nciSatConfig3	Satellite Configuration Property 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
49	mnc5	nciSatConfig4	Satellite Configuration Property 4 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
50	mnc6	nciSatConfig5	Satellite Configuration Property 5 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
51	mnc7	nciSatConfig6	Satellite Configuration Property 6 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
52	mnc8	nciSatConfig7	Satellite Configuration Property 7 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
53	mnc9	nciSatConfig8	Satellite Configuration Property 8 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

Table–B.4 Invensys Satellite 1 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:10. (Continued)

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nci*Name*	**Invensys Configuration Properties (continued)**						
54	mnc10	nciSatConfig9	Satellite Configuration Property 9 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
55	mnc11	nciSatConfig10	Satellite Configuration Property 10 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
56	mnc18	nciSatConfig11	Satellite Configuration Property 11 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
57	mnc19	nciSatConfig12	Satellite Configuration Property 12 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
58	mnc2	nciSECModelNum	Controller Part Number and Application Number	SNVT_str_asc	36	1Not Supported		

Satellite 2 Profile

The Invensys Satellite 2 profile is graphically shown below Figure-3.



Note: Version 1, MNL-xxRS3 Controllers use Program ID 80:00:16:50:1E:04:04:11.

Figure–B.5 Satellite 2 Profile

Satellite 2 Profile Quick Reference

For each Satellite 2 profile item, Table–B.7 includes the network variable (NV) index number, self-doc index number, item name, general description, SNVT type, element names and numbers as applicable, and point type (AV = Analog Value, DV = Digital Value). An example Profile tag shape is shown in the header above each class of profile items.

Table–B.5 Invensys Satellite 2 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:11.

NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nviName	**Mandatory Network Variable Inputs**						
0	nv1	nviSpaceTemp	Space Temperature (In)	SNVT_temp_p	105	—	—	AV
1	nv2	nviSetPoint	Single Setpoint	SNVT_temp_p	105	—	—	AV
	nvoName	**Mandatory Network Variable Outputs**						
2	nv3	nvoSpaceTemp	Space Temperature (Out)	SNVT_temp_p	105	—	—	AV
3	nv4	nvoUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode	1	AV
						.Heat2	2	AV
						.Heat1	3	AV
						.Cool	4	AV
						.Econ	5	AV
						.Fan	6	AV
						.Alarm	7	AV
	nviName	**Optional Network Variable Inputs**						
4	nv5	nviApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
5	nv6	nviOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
—	nv71	nviSetPtOffset1	Single Setpoint Offset1	SNVT_temp_p	105	1Not Supported		
—	nv81	nviOutsideTemp1	Outside Air Temperature (In)1	SNVT_temp_p	105	1Not Supported		
—	nv91	nviOutsideRH1	Outside Air Relative Humidity (In)1	SNVT_lev_percent	81	1Not Supported		
—	nv131	nviSpaceRH1	Space Relative Humidity (In)1	SNVT_lev_percent	81	1Not Supported		
—	nv141	nviCO21	CO2 Sensor in PPM (In)1	SNVT_ppm	29	1Not Supported		
—	nv151	nviEmergCmd1	Emergency Command (In)1	SNVT_hvac_emerg	103	1Not Supported		
	nvoName	**Optional Network Variable Outputs**						
—	nv101	nvoEffectSetPt1	Setpoint Value Used1	SNVT_temp_p	105	1Not Supported		
—	nv111	nvoOutsideTemp1	Outside Air Temperature (Out)1	SNVT_temp_p	105	1Not Supported		
—	nv121	nvoOutsideRH1	Outside Air Relative Humidity (Out)1	SNVT_lev_percent	81	1Not Supported		
—	nv161	nvoCO21	CO2 Sensor in PPM (Out)1	SNVT_ppm	29	1Not Supported		
	nciName	**Configuration Properties**						
45	nc60	nciSetPnts	Default Setpoints	SNVT_temp_setpt	106	.OccCool	1	AV
						.SBCool	2	AV
						.UnoccCool	3	AV
						.OccHeat	4	AV
						.SBHeat	5	AV
						.UnoccHeat	6	AV
46	nc49	nciSndHrtBt	Send Heartbeat (see page 547)	SNVT_time_sec	107	—	—	AV
47	nc48	nciRcvHrtBt	Receive Heartbeat (see page 545)	SNVT_time_sec	107	—	—	AV
48	nc17	nciLocation	Text Descriptor for Physical Location	SNVT_str_asc	36	1Not Supported		
—	nc421	nciCO2Limit1	CO2 Level Limit1	SNVT_ppm	29	1Not Supported		

Table–B.5 Invensys Satellite 2 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:11. (Continued)

NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs**						
6	mnv1	nviTimeStamp	Time Input from External Device *Note:* This NVI is used to sync the controller clock; no usable NVI tags are in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
7	mnv2	nviOccSchedule	Occupancy Scheduling Input	SNVT_tod_event	128	.Current .Next .Time	1 2 3	AV AV AV
8	mnv3	nviOverride	Override Input	SNVT_hvac_overid	111	.State .Percent .Flow	1 2 3	AV AV AV
9	mnv4	nviSatTemp1	Satellite Temperature Input 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
10	mnv5	nviSatTemp2	Satellite Temperature Input 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
11	mnv6	nviSatERTemp1	Satellite Extended Range (ER) Temperature Input 1 (defined by the application)	SNVT_temp	39	—	—	AV
12	mnv7	nviSatPercent1	Satellite Percent Input 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
13	mnv8	nviSatPercent2	Satellite Percent Input 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
14	mnv9	nviSatSwitch1	Satellite Switch Input 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
15	mnv10	nviSatSwitch2	Satellite Switch Input 2 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
16	mnv11	nviSatLevDisc1	Satellite Level Discrete Input 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
17	mnv12	nviSatCntIncF1	Satellite Count Incremental Input 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
18	mnv13	nviSatCntIncF2	Satellite Count Incremental Input 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
19	mnv14	nviSatFlow1	Satellite Flow Input	SNVT_flow	15	—	—	AV
20	mnv15	nviUnitStatus	Current Status of Unit Input	SNVT_hvac_status	112	.Mode .Heat2 .Heat1 .Cool .Econ .Fan .Alarm	1 2 3 4 5 6 7	AV AV AV AV AV AV AV
21	mnv16	nviSatPPM1	Satellite PPM Input 1 (defined by the application)	SNVT_ppm	29	—	—	AV
22	mnv17	nviSatPPM2	Satellite PPM Input 2 (defined by the application)	SNVT_ppm	29	—	—	AV

Table–B.5 Invensys Satellite 2 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:11. (Continued)

NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs (continued)**						
23	mnv18	nviSatPressP1	Satellite Pressure Input 1	SNVT_press_p	113	—	—	AV
24	mnv19	nviSatState1	Satellite State Input 1 (defined by the application)	SNVT_state	83	.State0	1	DV
						.State1	2	DV
						.State2	3	DV
						.State3	4	DV
						.State4	5	DV
						.State5	6	DV
						.State6	7	DV
						.State7	8	DV
						.State8	9	DV
						.State9	10	DV
						.State10	11	DV
						.State11	12	DV
						.State12	13	DV
						.State13	14	DV
						.State14	15	DV
						.State15	16	DV
	nvo*Name*	**Invensys Network Variable Outputs**						
25	mnv20	nvoSECAlarm	Alarm Output (Future Use, No WP Tech visibility)	SNVT_str_asc (SEC_alarm if revision 1 profile)	36	1Not Supported		
26	mnv21	nvoTimeStamp	Time Output *Note:* This NVO transmits the time and date of the controller clock. No usable NVO tags are in WP Tech.	SNVT_time_stamp	84	(See *Note*)		
						.Year	1	AV
						.Month	2	AV
						.Day	3	AV
						.Hour	4	AV
						.Minute	5	AV
						.Second	6	AV
27	mnv22	nvoOccSchedule	Occupancy Scheduling Output	SNVT_tod_event	128	.Current	1	AV
						.Next	2	AV
						.Time	3	AV
28	mnv23	nvoOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
29	mnv24	nvoSatTemp1	Satellite Temperature Output 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
30	mnv25	nvoSatTemp2	Satellite Temperature Output 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
31	mnv26	nvoSatERTemp1	Satellite Extended Range (ER) Temperature Output 1 (defined by the application)	SNVT_temp	39	—	—	AV
32	mnv27	nvoSatPercent1	Satellite Percent Output 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
33	mnv28	nvoSatPercent2	Satellite Percent Output 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
34	mnv29	nvoSatSwitch1	Satellite Switch Output 1 (defined by the application)	SNVT_switch	95	.Value	1	AV
						.State	2	DV

Table–B.5 Invensys Satellite 2 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:11. (Continued)

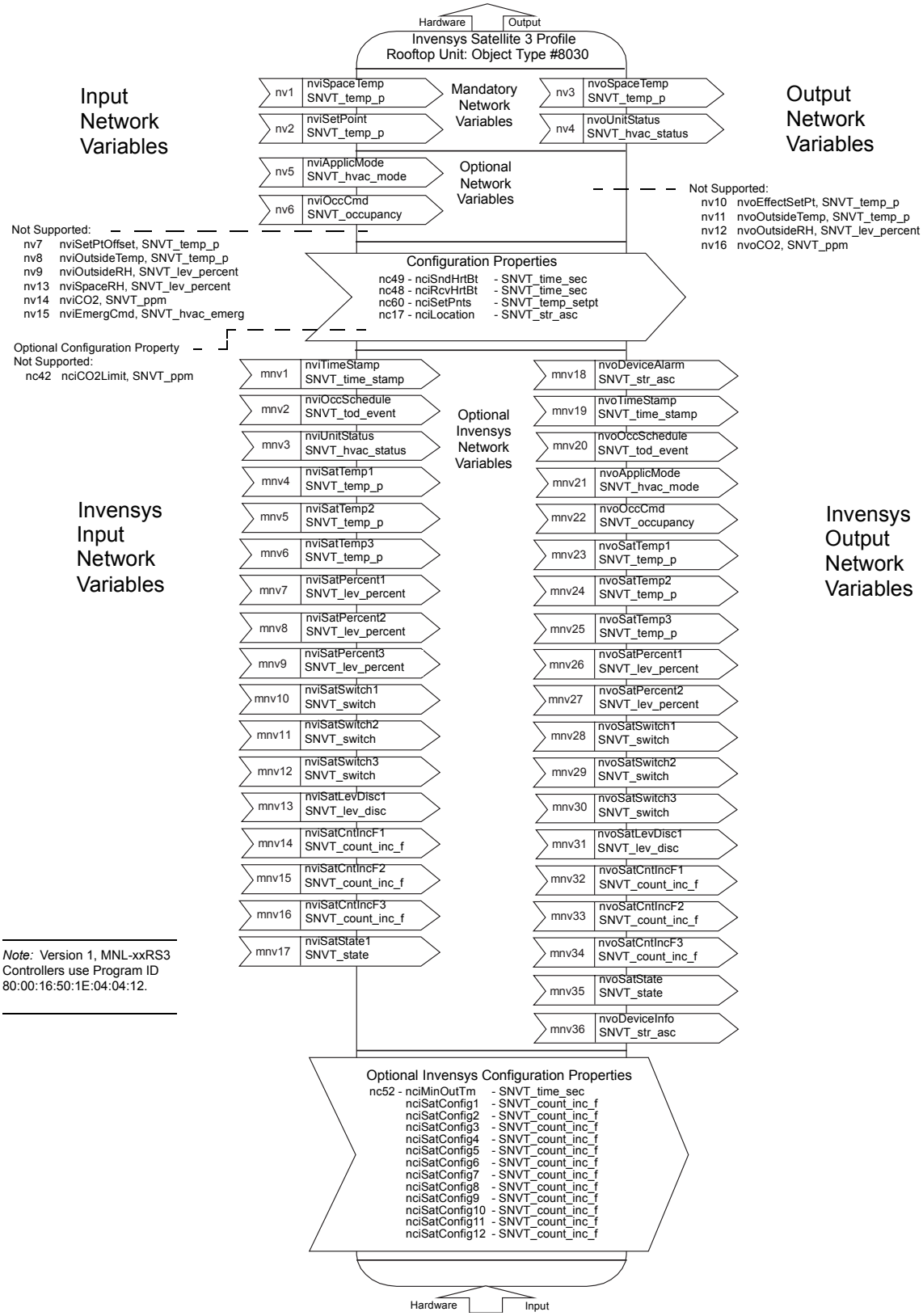
NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvo*Name*	**Invensys Network Variable Outputs (continued)**						
35	mnv30	nvoSatSwitch2	Satellite Switch Output 2 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
36	mnv31	nvoSatLevDisc1	Satellite Level Discrete Output 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
37	mnv32	nvoSatCntIncF1	Satellite Count Incremental Output 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
38	mnv33	nvoSatCntIncF2	Satellite Count Incremental Output 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
39	mnv34	nvoSatFlow1	Satellite FlowOutput 1 (defined by the application)	SNVT_flow	15	—	—	AV
40	mnv35	nvoApplicMode	Satellite Mode Command (defined by the application)	SNVT_hvac_mode	108	—	—	AV
41	mnv36	nvoOverride	Satellite Override Output (defined by the application)	SNVT_hvac_overid	111	.State .Percent .Flow	1 2 3	AV AV AV
42	mnv37	nvoSatPPM1	Satellite PPM Output 1 (defined by the application)	SNVT_ppm	29	—	—	AV
43	mnv38	nvoSatPressP1	Satellite Pressure Output 1	SNVT_press_p	113	—	—	AV
44	mnv35	nvoSatState1	Satellite State Output 1 (defined by the application)	SNVT_state	83	.State0 .State1 .State2 .State3 .State4 .State5 .State6 .State7 .State8 .State9 .State10 .State11 .State12 .State13 .State14 .State15	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	DV DV DV DV DV DV DV DV DV DV DV DV DV DV DV DV
	nci*Name*	**Invensys Configuration Properties**						
49	nc52	nciMinOutTm	Minimum Time between Output Transmissions (see page 547)	SNVT_time_sec	107	—	—	AV
50	mnc1	nciSatConfig1	Satellite Configuration Property 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
51	mnc3	nciSatConfig2	Satellite Configuration Property 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
52	mnc4	nciSatConfig3	Satellite Configuration Property 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
53	mnc5	nciSatConfig4	Satellite Configuration Property 4 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

Table–B.5 Invensys Satellite 2 (Roof Top Unit) Profile Items, Program ID 80:00:16:50:1E:04:04:11. (Continued)

NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nci*Name*	**Invensys Configuration Properties (continued)**						
54	mnc6	nciSatConfig5	Satellite Configuration Property 5 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
55	mnc7	nciSatConfig6	Satellite Configuration Property 6 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
56	mnc8	nciSatConfig7	Satellite Configuration Property 7 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
57	mnc9	nciSatConfig8	Satellite Configuration Property 8 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
58	mnc10	nciSatConfig9	Satellite Configuration Property 9 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
59	mnc11	nciSatConfig10	Satellite Configuration Property 10 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
60	mnc2	nciSECModelNum	Controller Part Number and Application Number	SNVT_str_asc	36	1Not Supported		

Satellite 3 Profile

The Invensys Satellite 3 profile is graphically shown below Figure-3.



Note: Version 1, MNL-xxRS3 Controllers use Program ID 80:00:16:50:1E:04:04:12.

Figure–B.6 Satellite 3 Profile

Satellite 3 Profile Quick Reference

For each Satellite 3 profile item, Table–B.6 includes the network variable (NV) index number, self-doc index number, item name, general description, SNVT type, element names and numbers as applicable, and point type (AV = Analog Value, DV = Digital Value). An example Profile tag shape is shown in the header above each class of profile items.

Table–B.6 Invensys Satellite 3 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:12.

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nviName	**Mandatory Network Variable Inputs**						
0	nv1	nviSpaceTemp	Space Temperature (In)	SNVT_temp_p	105	—	—	AV
1	nv2	nviSetPoint	Single Setpoint	SNVT_temp_p	105	—	—	AV
	nvoName	**Mandatory Network Variable Outputs**						
2	nv3	nvoSpaceTemp	Space Temperature (Out)	SNVT_temp_p	105	—	—	AV
3	nv4	nvoUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode	1	AV
						.Heat2	2	AV
						.Heat1	3	AV
						.Cool	4	AV
						.Econ	5	AV
						.Fan	6	AV
						.Alarm	7	AV
	nviName	**Optional Network Variable Inputs**						
4	nv5	nviApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
5	nv6	nviOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
—	nv71	nviSetPtOffset1	Single Setpoint Offset1	SNVT_temp_p	105	1Not Supported		
—	nv81	nviOutsideTemp1	Outside Air Temperature (In)1	SNVT_temp_p	105	1Not Supported		
—	nv91	nviOutsideRH1	Outside Air Relative Humidity (In)1	SNVT_lev_percent	81	1Not Supported		
—	nv131	nviSpaceRH1	Space Relative Humidity (In)1	SNVT_lev_percent	81	1Not Supported		
—	nv141	nviCO21	CO2 Sensor in PPM (In)1	SNVT_ppm	29	1Not Supported		
—	nv151	nviEmergCmd1	Emergency Command (In)1	SNVT_hvac_emerg	103	1Not Supported		
	nvoName	**Optional Network Variable Outputs**						
—	nv101	nvoEffectSetPt1	Setpoint Value Used1	SNVT_temp_p	105	1Not Supported		
—	nv111	nvoOutsideTemp1	Outside Air Temperature (Out)1	SNVT_temp_p	105	1Not Supported		
—	nv121	nvoOutsideRH1	Outside Air Relative Humidity (Out)1	SNVT_lev_percent	81	1Not Supported		
—	nv161	nvoCO21	CO2 Sensor in PPM (Out)1	SNVT_ppm	29	1Not Supported		
	nciName	**Configuration Properties**						
41	nc60	nciSetPnts	Default Setpoints	SNVT_temp_setpt	106	.OccCool	1	AV
						.SBCool	2	AV
						.UnoccCool	3	AV
						.OccHeat	4	AV
						.SBHeat	5	AV
						.UnoccHeat	6	AV
42	nc49	nciSndHrtBt	Send Heartbeat (see page 547)	SNVT_time_sec	107	—	—	AV
43	nc48	nciRcvHrtBt	Receive Heartbeat (see page 545)	SNVT_time_sec	107	—	—	AV
44	nc17	nciLocation	Text Descriptor for Physical Location	SNVT_str_asc	36	1Not Supported		
—	nc421	nciCO2Limit1	CO2 Level Limit1	SNVT_ppm	29	1Not Supported		

Table–B.6 Invensys Satellite 3 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:12. (Continued)

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs**						
6	mnv1	nviTimeStamp	Time Input from External Device *Note:* This NVI is used to sync the controller clock; no usable NVI tags are in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
7	mnv2	nviOccSchedule	Occupancy Scheduling Input	SNVT_tod_event	128	.Current .Next .Time	1 2 3	AV AV AV
8	mnv3	nviUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode .Heat2 .Heat1 .Cool .Econ .Fan .Alarm	1 2 3 4 5 6 7	AV AV AV AV AV AV AV
9	mnv4	nviSatTemp1	Satellite Temperature Input 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
10	mnv5	nviSatTemp2	Satellite Temperature Input 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
11	mnv6	nviSatTemp3	Satellite Temperature Input 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
12	mnv7	nviSatPercent1	Satellite Percent Input 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
13	mnv8	nviSatPercent2	Satellite Percent Input 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
14	mnv9	nviSatPercent3	Satellite Percent Input 3 (defined by the application)	SNVT_lev_percent	81	—	—	AV
15	mnv10	nviSatSwitch1	Satellite Switch Input 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
16	mnv11	nviSatSwitch2	Satellite Switch Input 2 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
17	mnv12	nviSatSwitch3	Satellite Switch Input 3 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
18	mnv13	nviSatLevDisc1	Satellite Level Discrete Input 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
19	mnv14	nviSatCntIncF1	Satellite Count Incremental Input 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
20	mnv15	nviSatCntIncF2	Satellite Count Incremental Input 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
21	mnv16	nviSatCntIncF3	Satellite Count Incremental Input 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

Table–B.6 Invensys Satellite 3 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:12. (Continued)

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs (continued)**						
22	mnv17	nviSatState1	Satellite State Input 1 (defined by the application)	SNVT_state	83	.State0	1	DV
						.State1	2	DV
						.State2	3	DV
						.State3	4	DV
						.State4	5	DV
						.State5	6	DV
						.State6	7	DV
						.State7	8	DV
						.State8	9	DV
						.State9	10	DV
						.State10	11	DV
						.State11	12	DV
						.State12	13	DV
						.State13	14	DV
						.State14	15	DV
						.State15	16	DV
	nvo*Name*	**Invensys Network Variable Outputs**						
23	mnv18	nvoDeviceAlarm	Alarm Output (Future Use, No WP Tech visibility)	SNVT_str_asc	36	1Not Supported		
24	mnv19	nvoTimeStamp	Time Output *Note:* This NVO transmits the time and date of the controller clock. No usable NVO tags are in WP Tech.	SNVT_time_stamp	84	(See *Note*)		
						.Year	1	AV
						.Month	2	AV
						.Day	3	AV
						.Hour	4	AV
						.Minute	5	AV
						.Second	6	AV
25	mnv20	nvoOccSchedule	Occupancy Scheduling Output	SNVT_tod_event	128	.Current	1	AV
						.Next	2	AV
						.Time	3	AV
26	mnv21	nvoApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
27	mnv22	nvoOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
28	mnv23	nvoSatTemp1	Satellite Temperature Output 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
29	mnv24	nvoSatTemp2	Satellite Temperature Output 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
30	mnv25	nvoSatTemp3	Satellite Temperature Output 3 (defined by the application)	SNVT_temp_p	105	—	—	AV
31	mnv26	nvoSatPercent1	Satellite Percent Output 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
32	mnv27	nvoSatPercent2	Satellite Percent Output 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
33	mnv28	nvoSatSwitch1	Satellite Switch Output 1 (defined by the application)	SNVT_switch	95	.Value	1	AV
						.State	2	DV
34	mnv29	nvoSatSwitch2	Satellite Switch Output 2 (defined by the application)	SNVT_switch	95	.Value	1	AV
						.State	2	DV

Table–B.6 Invensys Satellite 3 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:12. (Continued)

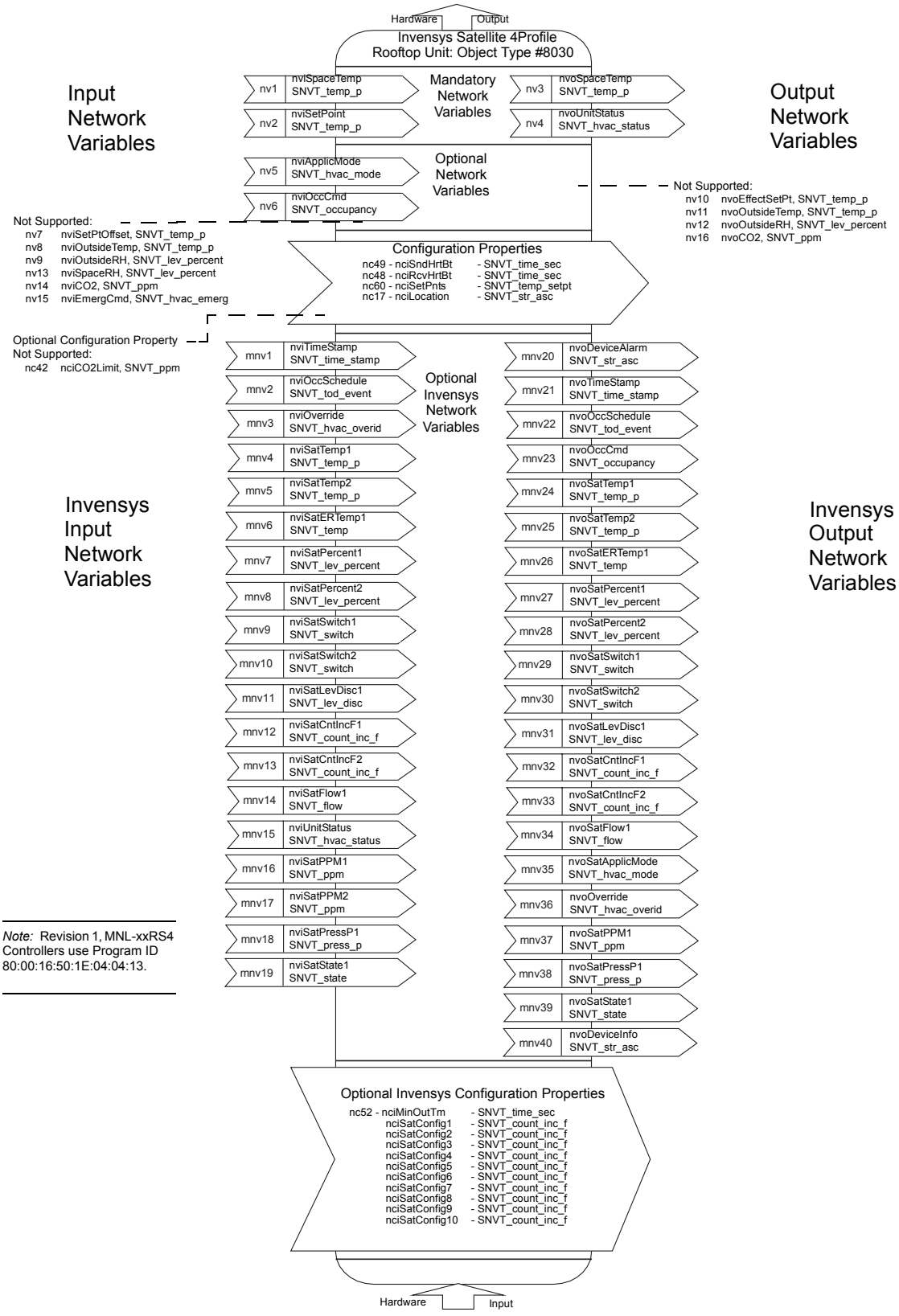
NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvo*Name*	**Invensys Network Variable Outputs (continued)**						
35	mnv30	nvoSatSwitch3	Satellite Switch Output 3 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
36	mnv31	nvoSatLevDisc1	Satellite Level Discrete Output 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
37	mnv32	nvoSatCntIncF1	Satellite Count Incremental Output 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
38	mnv33	nvoSatCntIncF2	Satellite Count Incremental Output 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
39	mnv34	nvoSatCntIncF3	Satellite Count Incremental Output 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
40	mnv35	nvoSatState1	Satellite State Output 1 (defined by the application)	SNVT_state	83	.State0 .State1 .State2 .State3 .State4 .State5 .State6 .State7 .State8 .State9 .State10 .State11 .State12 .State13 .State14 .State15	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	DV DV DV DV DV DV DV DV DV DV DV DV DV DV DV DV
58	mnv36	nvoDeviceInfo	Controller Model Number, Firmware Version, Diagnostic Status, and Application Name	SNVT_str_asc	36	1Not Supported		
	nci*Name*	**Invensys Configuration Properties**						
45	nc52	nciMinOutTm	Minimum Time between Output Transmissions (see page 547)	SNVT_time_sec	107	—	—	AV
46	mnc1	nciSatConfig1	Satellite Configuration Property 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
47	mnc3	nciSatConfig2	Satellite Configuration Property 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
48	mnc4	nciSatConfig3	Satellite Configuration Property 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
49	mnc5	nciSatConfig4	Satellite Configuration Property 4 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
50	mnc6	nciSatConfig5	Satellite Configuration Property 5 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
51	mnc7	nciSatConfig6	Satellite Configuration Property 6 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
52	mnc8	nciSatConfig7	Satellite Configuration Property 7 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

Table–B.6 Invensys Satellite 3 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:12. (Continued)

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nci*Name*	**Invensys Configuration Properties (continued)**						
53	mnc9	nciSatConfig8	Satellite Configuration Property 8 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
54	mnc10	nciSatConfig9	Satellite Configuration Property 9 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
55	mnc11	nciSatConfig10	Satellite Configuration Property 10 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
56	mnc18	nciSatConfig11	Satellite Configuration Property 11 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
57	mnc19	nciSatConfig12	Satellite Configuration Property 12 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

Satellite 4 Profile

The Invensys Satellite 4 profile is graphically shown below Figure-3.



Note: Revision 1, MNL-xxRS4 Controllers use Program ID 80:00:16:50:1E:04:04:13.

Figure–B.7 Satellite 4 Profile

Satellite 4 Profile Quick Reference

For each Satellite 4 profile item, Table–B.7 includes the network variable (NV) index number, self-doc index number, item name, general description, SNVT type, element names and numbers as applicable, and point type (AV = Analog Value, DV = Digital Value). An example Profile tag shape is shown in the header above each class of profile items.

Table–B.7 Invensys Satellite 4 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:13.

NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nviName	**Mandatory Network Variable Inputs**						
0	nv1	nviSpaceTemp	Space Temperature (In)	SNVT_temp_p	105	—	—	AV
1	nv2	nviSetPoint	Single Setpoint	SNVT_temp_p	105	—	—	AV
	nvoName	**Mandatory Network Variable Outputs**						
2	nv3	nvoSpaceTemp	Space Temperature (Out)	SNVT_temp_p	105	—	—	AV
3	nv4	nvoUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode	1	AV
						.Heat2	2	AV
						.Heat1	3	AV
						.Cool	4	AV
						.Econ	5	AV
						.Fan	6	AV
						.Alarm	7	AV
	nviName	**Optional Network Variable Inputs**						
4	nv5	nviApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
5	nv6	nviOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
—	nv71	nviSetPtOffset1	Single Setpoint Offset1	SNVT_temp_p	105	1Not Supported		
—	nv81	nviOutsideTemp1	Outside Air Temperature (In)1	SNVT_temp_p	105	1Not Supported		
—	nv91	nviOutsideRH1	Outside Air Relative Humidity (In)1	SNVT_lev_percent	81	1Not Supported		
—	nv131	nviSpaceRH1	Space Relative Humidity (In)1	SNVT_lev_percent	81	1Not Supported		
—	nv141	nviCO21	CO2 Sensor in PPM (In)1	SNVT_ppm	29	1Not Supported		
—	nv151	nviEmergCmd1	Emergency Command (In)1	SNVT_hvac_emerg	103	1Not Supported		
	nvoName	**Optional Network Variable Outputs**						
—	nv101	nvoEffectSetPt1	Setpoint Value Used1	SNVT_temp_p	105	1Not Supported		
—	nv111	nvoOutsideTemp1	Outside Air Temperature (Out)1	SNVT_temp_p	105	1Not Supported		
—	nv121	nvoOutsideRH1	Outside Air Relative Humidity (Out)1	SNVT_lev_percent	81	1Not Supported		
—	nv161	nvoCO21	CO2 Sensor in PPM (Out)1	SNVT_ppm	29	1Not Supported		
	nciName	**Configuration Properties**						
45	nc60	nciSetPnts	Default Setpoints	SNVT_temp_setpt	106	.OccCool	1	AV
						.SBCool	2	AV
						.UnoccCool	3	AV
						.OccHeat	4	AV
						.SBHeat	5	AV
						.UnoccHeat	6	AV
46	nc49	nciSndHrtBt	Send Heartbeat (see page 547)	SNVT_time_sec	107	—	—	AV
47	nc48	nciRcvHrtBt	Receive Heartbeat (see page 545)	SNVT_time_sec	107	—	—	AV
48	nc17	nciLocation	Text Descriptor for Physical Location	SNVT_str_asc	36	1Not Supported		
—	nc421	nciCO2Limit1	CO2 Level Limit1	SNVT_ppm	29	1Not Supported		

Table–B.7 Invensys Satellite 4 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:13. (Continued)

NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs**						
6	mnv1	nviTimeStamp	Time Input from External Device *Note:* This NVI is used to sync the controller clock; no usable NVI tags are in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
7	mnv2	nviOccSchedule	Occupancy Scheduling Input	SNVT_tod_event	128	.Current .Next .Time	1 2 3	AV AV AV
8	mnv3	nviOverride	Override Input	SNVT_hvac_overid	111	.State .Percent .Flow	1 2 3	AV AV AV
9	mnv4	nviSatTemp1	Satellite Temperature Input 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
10	mnv5	nviSatTemp2	Satellite Temperature Input 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
11	mnv6	nviSatERTemp1	Satellite Extended Range (ER) Temperature Input 1 (defined by the application)	SNVT_temp	39	—	—	AV
12	mnv7	nviSatPercent1	Satellite Percent Input 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
13	mnv8	nviSatPercent2	Satellite Percent Input 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
14	mnv9	nviSatSwitch1	Satellite Switch Input 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
15	mnv10	nviSatSwitch2	Satellite Switch Input 2 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
16	mnv11	nviSatLevDisc1	Satellite Level Discrete Input 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
17	mnv12	nviSatCntIncF1	Satellite Count Incremental Input 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
18	mnv13	nviSatCntIncF2	Satellite Count Incremental Input 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
19	mnv14	nviSatFlow1	Satellite Flow Input	SNVT_flow	15	—	—	AV
20	mnv15	nviUnitStatus	Current Status of Unit Input	SNVT_hvac_status	112	.Mode .Heat2 .Heat1 .Cool .Econ .Fan .Alarm	1 2 3 4 5 6 7	AV AV AV AV AV AV AV
21	mnv16	nviSatPPM1	Satellite PPM Input 1 (defined by the application)	SNVT_ppm	29	—	—	AV
22	mnv17	nviSatPPM2	Satellite PPM Input 2 (defined by the application)	SNVT_ppm	29	—	—	AV

Table–B.7 Invensys Satellite 4 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:13. (Continued)

NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Invensys Network Variable Inputs (continued)**						
23	mnv18	nviSatPressP1	Satellite Pressure Input 1	SNVT_press_p	113	—	—	AV
24	mnv19	nviSatState1	Satellite State Input 1 (defined by the application)	SNVT_state	83	.State0	1	DV
						.State1	2	DV
						.State2	3	DV
						.State3	4	DV
						.State4	5	DV
						.State5	6	DV
						.State6	7	DV
						.State7	8	DV
						.State8	9	DV
						.State9	10	DV
						.State10	11	DV
						.State11	12	DV
						.State12	13	DV
						.State13	14	DV
						.State14	15	DV
						.State15	16	DV
	nvo*Name*	**Invensys Network Variable Outputs**						
25	mnv20	nvoDeviceAlarm	Alarm Output (Future Use, No WP Tech visibility)	SNVT_str_asc	36	1Not Supported		
26	mnv21	nvoTimeStamp	Time Output *Note:* This NVO transmits the time and date of the controller clock. No usable NVO tags are in WP Tech.	SNVT_time_stamp	84	(See *Note*)		
						.Year	1	AV
						.Month	2	AV
						.Day	3	AV
						.Hour	4	AV
						.Minute	5	AV
						.Second	6	AV
27	mnv22	nvoOccSchedule	Occupancy Scheduling Output	SNVT_tod_event	128	.Current	1	AV
						.Next	2	AV
						.Time	3	AV
28	mnv23	nvoOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
29	mnv24	nvoSatTemp1	Satellite Temperature Output 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
30	mnv25	nvoSatTemp2	Satellite Temperature Output 2 (defined by the application)	SNVT_temp_p	105	—	—	AV
31	mnv26	nvoSatERTemp1	Satellite Extended Range (ER) Temperature Output 1 (defined by the application)	SNVT_temp	39	—	—	AV
32	mnv27	nvoSatPercent1	Satellite Percent Output 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
33	mnv28	nvoSatPercent2	Satellite Percent Output 2 (defined by the application)	SNVT_lev_percent	81	—	—	AV
34	mnv29	nvoSatSwitch1	Satellite Switch Output 1 (defined by the application)	SNVT_switch	95	.Value	1	AV
						.State	2	DV

Table–B.7 Invensys Satellite 4 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:13. (Continued)

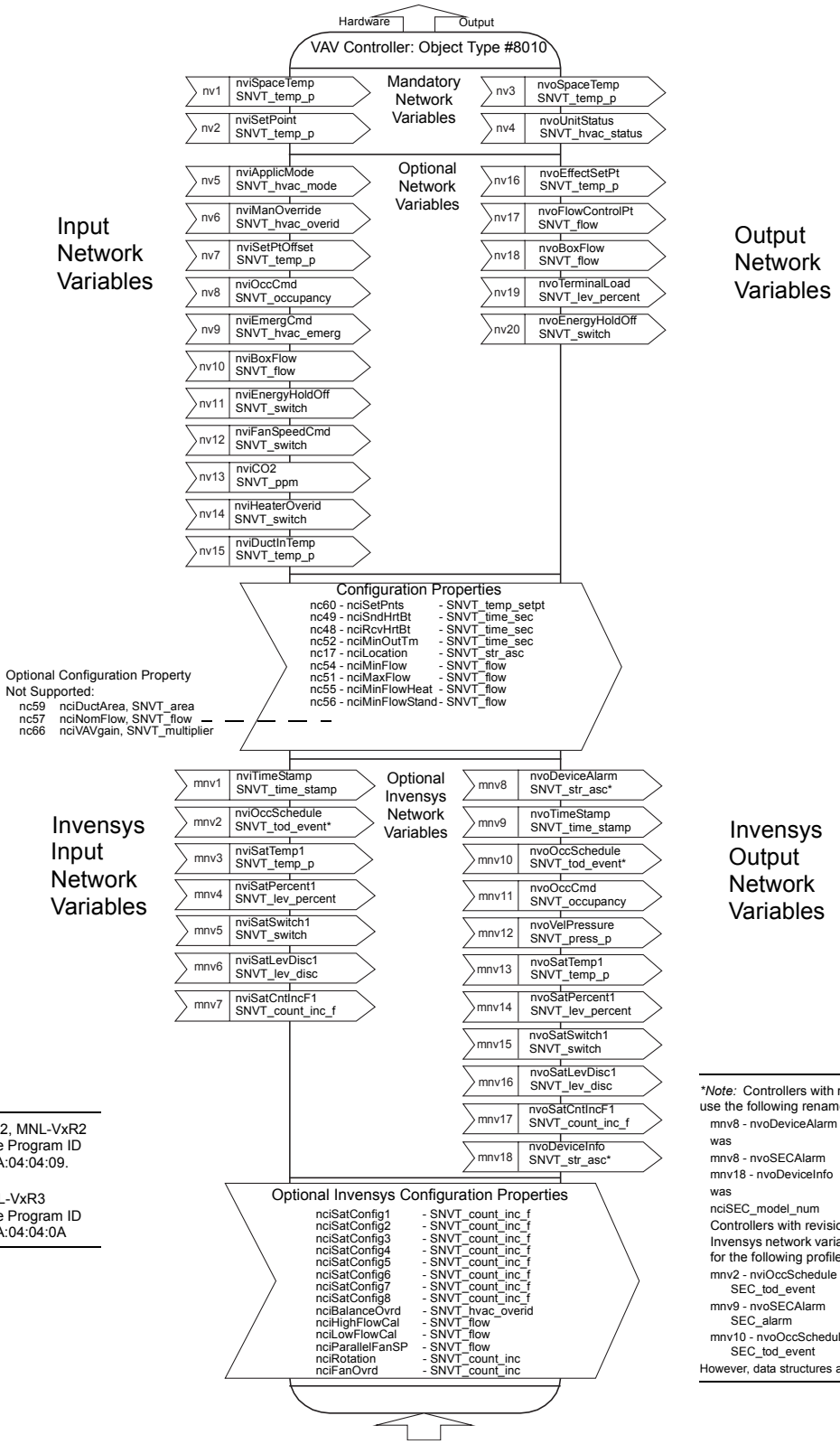
NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvo*Name*	**Invensys Network Variable Outputs (continued)**						
35	mnv30	nvoSatSwitch2	Satellite Switch Output 2 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
36	mnv31	nvoSatLevDisc1	Satellite Level Discrete Output 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
37	mnv32	nvoSatCntIncF1	Satellite Count Incremental Output 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
38	mnv33	nvoSatCntIncF2	Satellite Count Incremental Output 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
39	mnv34	nvoSatFlow1	Satellite FlowOutput 1 (defined by the application)	SNVT_flow	15	—	—	AV
40	mnv35	nvoApplicMode	Satellite Mode Command (defined by the application)	SNVT_hvac_mode	108	—	—	AV
41	mnv36	nvoOverride	Satellite Override Output (defined by the application)	SNVT_hvac_overid	111	.State .Percent .Flow	1 2 3	AV AV AV
42	mnv37	nvoSatPPM1	Satellite PPM Output 1 (defined by the application)	SNVT_ppm	29	—	—	AV
43	mnv38	nvoSatPressP1	Satellite Pressure Output 1	SNVT_press_p	113	—	—	AV
44	mnv39	nvoSatState1	Satellite State Output 1 (defined by the application)	SNVT_state	83	.State0 .State1 .State2 .State3 .State4 .State5 .State6 .State7 .State8 .State9 .State10 .State11 .State12 .State13 .State14 .State15	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16	DV DV DV DV DV DV DV DV DV DV DV DV DV DV DV DV
60	mnv40	nvoDeviceInfo	Controller Model Number, Firmware Version, Diagnostic Status, and Application Name	SNVT_str_asc	36	1Not Supported		
	nci*Name*	**Invensys Configuration Properties**						
49	nc52	nciMinOutTm	Minimum Time between Output Transmissions (see page 547)	SNVT_time_sec	107	—	—	AV
50	mnc1	nciSatConfig1	Satellite Configuration Property 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
51	mnc3	nciSatConfig2	Satellite Configuration Property 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
52	mnc4	nciSatConfig3	Satellite Configuration Property 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

Table–B.7 Invensys Satellite 4 (Roof Top Unit) Profile Items: Program ID 80:00:16:50:1E:04:04:13. (Continued)

NV index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nci*Name*	**Invensys Configuration Properties (continued)**						
53	mnc5	nciSatConfig4	Satellite Configuration Property 4 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
54	mnc6	nciSatConfig5	Satellite Configuration Property 5 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
55	mnc7	nciSatConfig6	Satellite Configuration Property 6 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
56	mnc8	nciSatConfig7	Satellite Configuration Property 7 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
57	mnc9	nciSatConfig8	Satellite Configuration Property 8 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
58	mnc10	nciSatConfig9	Satellite Configuration Property 9 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
59	mnc11	nciSatConfig10	Satellite Configuration Property 10 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

VAV Controller Profile

The Invensys VAV Controller profile is graphically shown below Figure-8.



Note: Version 2, MNL-VxR2 Controllers use Program ID 80:00:16:50:0A:04:04:09.

Version 3, MNL-VxR3 Controllers use Program ID 80:00:16:50:0A:04:04:0A

Note: Controllers with revision 3 VAV profiles use the following renamed profile items:

mnv8 - nvoDeviceAlarm
was
mnv8 - nvoSECAlarm
mnv18 - nvoDeviceInfo
was
nciSEC_model_num

Controllers with revision 1 VAV profiles use Invensys network variable types (vs. SNVTs) for the following profile items:

mnv2 - nviOccSchedule
SEC_tod_event
mnv9 - nvoSECAlarm
SEC_alarm
mnv10 - nvoOccSchedule
SEC_tod_event

However, data structures are equivalent.

Figure–B.8 VAV Controller Profile

VAV Controller Profile Quick Reference

For each VAV controller profile item, Table–B.8 includes the network variable (NV) index number, self-doc index number, item name, general description, SNVT type, element names and numbers as applicable, and point type (AV = Analog Value, DV = Digital Value). An example Profile tag shape is shown in the header above each class of profile items.

Table–B.8 Invensys LONMARK VAV Profile Items: Version 2 - Program ID 80:00:16:50:0A:04:04:09 and Version 3 - Program ID 80:00:16:50:0A:04:04:0A

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvi*Name*	**Mandatory Network Variable Inputs**						
0	nv1	nviSpaceTemp	Space Temperature (In)	SNVT_temp_p	105	—	—	AV
1	nv2	nviSetPoint	Single Setpoint	SNVT_temp_p	105	—	—	AV
	nvo*Name*	**Mandatory Network Variable Outputs**						
2	nv3	nvoSpaceTemp	Space Temperature (Out)	SNVT_temp_p	105	—	—	AV
3	nv4	nvoUnitStatus	Current Status of Unit	SNVT_hvac_status	112	.Mode	1	AV
						.Heat2	2	AV
						.Heat1	3	AV
						.Cool	4	AV
						.Econ	5	AV
						.Fan	6	AV
						.Alarm	7	AV
	nvi*Name*	**Optional Network Variable Inputs**						
4	nv5	nviApplicMode	Mode Command	SNVT_hvac_mode	108	—	—	AV
5	nv6	nviManOverride	Manual Override	SNVT_hvac_overid	111	.State	1	AV
						.Percent	2	AV
						.Flow	3	AV
6	nv7	nviSetPtOffset	Single Setpoint Offset	SNVT_temp_p	105	—	—	AV
7	nv8	nviOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
8	nv9	nviEmergCmd	Emergency Command (In)	SNVT_hvac_emerg	103	—	—	AV
9	nv10	nviBoxFlow	VAV Terminal Box Flow (In)	SNVT_flow	15	—	—	AV
10	nv11	nviEnergyHoldOff	Energy Hold Off Command	SNVT_switch	95	.Value	1	AV
						.State	2	DV
11	nv12	nviFanSpeedCmd	Fan Speed Command	SNVT_switch	95	.Value	1	AV
						.State	2	DV
12	nv13	nviCO2	CO2 Sensor in PPM (In)	SNVT_ppm	29	—	—	AV
13	nv14	nviHeaterOverd	Heater Override Command	SNVT_switch	95	.Value	1	AV
						.State	2	DV
14	nv15	nviDuctInTemp	Duct Temperature (In)	SNVT_temp_p	105	—	—	AV
	nvo*Name*	**Optional Network Variable Outputs**						
15	nv16	nvoEffectSetPt	Setpoint Value Used	SNVT_temp_p	105	—	—	AV
16	nv17	nvoFlowControlPt	Current Flow Control Value	SNVT_flow	15	—	—	AV
17	nv18	nvoBoxFlow	VAV Terminal Box Flow (Out)	SNVT_flow	15	—	—	AV
18	nv19	nvoTerminalLoad	VAV Terminal Load Level Percent	SNVT_lev_percent	81	—	—	AV
19	nv20	nvoEnergyHoldOff	State of Energy Hold-off Device Hardwired to Controller	SNVT_switch	95	.Value	1	AV
						.State	2	DV

Table–B.8 Invensys LONMARK VAV Profile Items: Version 2 - Program ID 80:00:16:50:0A:04:04:09 and Version 3 - Program ID 80:00:16:50:0A:04:04:0A

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nci*Name*	**Configuration Properties**						
37	nc60	nciSetPnts	Default Setpoints	SNVT_temp_setpt	106	.OccCool	1	AV
						.SBCool	2	AV
						.UnoccCool	3	AV
						.OccHeat	4	AV
						.SBHeat	5	AV
						.UnoccHeat	6	AV
38	nc49	nciSndHrtBt	Send Heartbeat (see page 547)	SNVT_time_sec	107	—	—	AV
39	nc48	nciRcvHrtBt	Receive Heartbeat (see page 545)	SNVT_time_sec	107	—	—	AV
40	nc52	nciMinOutTm	Minimum Time between Output Transmissions (see page 547)	SNVT_time_sec	107	—	—	AV
41	nc17	nciLocation	Text Descriptor for Physical Location	SNVT_str_asc	36	1Not Supported		
42	nc54	nciMinFlow	Minimum Flow	SNVT_flow	15	—		
43	nc51	nciMaxFlow	Maximum Flow	SNVT_flow	15	—		
44	nc55	nciMinFlowHeat	Minimum Flow for Heating	SNVT_flow	15	—		
45	nc56	nciMinFlowStand	Minimum Flow for Standby	SNVT_flow	15	—		
—	nc461	nciDuctArea1	Area of the Duct1	SNVT_area1	110	1Not Supported		
—	nc571	nciNomFlow1	Nominal Flow (least measurable)1	SNVT_flow	15	1Not Supported		
—	nc661	nciVAVgain1	Gain of the VAV Controller1	SNVT_multiplier1	82	1Not Supported		
	nvi*Name*	**Invensys Network Variable Inputs**						
20	mnv1	nviTimeStamp	Time Input from External Device *Note:* Rev.2 or higher profile uses this NVI to sync the controller clock, No usable NVI tags in WP Tech.	SNVT_time_stamp	84	(See *Note*)		
						.Year	1	AV
						.Month	2	AV
						.Day	3	AV
						.Hour	4	AV
						.Minute	5	AV
						.Second	6	AV
21	mnv2	nviOccSchedule	Occupancy Scheduling Input	SNVT_tod_event	128	.Current	1	AV
						.Next	2	AV
						.Time	3	AV
22	mnv3	nviSatTemp1	Satellite Temperature Input 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
23	mnv4	nviSatPercent1	Satellite Percent Input 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
24	mnv5	nviSatSwitch1	Satellite Switch Input 1 (defined by the application)	SNVT_switch	95	.Value	1	AV
						.State	2	DV
25	mnv6	nviSatLevDisc1	Satellite Level Discrete Input 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
26	mnv7	nviSatCntIncF1	Satellite Count Incremental Input 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

Table–B.8 Invensys LONMARK VAV Profile Items: Version 2 - Program ID 80:00:16:50:0A:04:04:09 and Version 3 - Program ID 80:00:16:50:0A:04:04:0A

NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nvo*Name*	**Invensys Network Variable Outputs**						
27	mnv8	nvoDeviceAlarm (nvoSECAlarm if version 1 or 2 profile)	Alarm Output (Future Use, No WP Tech visibility)	SNVT_str_asc (SEC_alarm if revision 1 profile)	36	1Not Supported		
28	mnv9	nvoTimeStamp	Time Output *Note:* Rev.2 or higher profile uses this NVO to transmit the time and date of the controller clock. No usable NVO tags in WP Tech.	SNVT_time_stamp	84	(See *Note*) .Year .Month .Day .Hour .Minute .Second	 1 2 3 4 5 6	 AV AV AV AV AV AV
29	mnv10	nvoOccSchedule	Occupancy Scheduling Output	SNVT_tod_event	128	.Current .Next .Time	1 2 3	AV AV AV
30	mnv11	nvoOccCmd	Occupancy Status (4-State)	SNVT_occupancy	109	—	—	AV
31	mnv12	nvoVelPressure	Velocity Pressure (Out)	SNVT_press_p	113	—	—	AV
32	mnv13	nvoSatTemp1	Satellite Temperature Output 1 (defined by the application)	SNVT_temp_p	105	—	—	AV
33	mnv14	nvoSatPercent1	Satellite Percent Output 1 (defined by the application)	SNVT_lev_percent	81	—	—	AV
34	mnv15	nvoSatSwitch1	Satellite Switch Output 1 (defined by the application)	SNVT_switch	95	.Value .State	1 2	AV DV
35	mnv16	nvoSatLevDisc1	Satellite Level Discrete Output 1 (defined by the application)	SNVT_lev_disc	22	—	—	AV
36	mnv17	nvoSatCntIncF1	Satellite Count Incremental Output 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
54[a]	mnv18	nvoDeviceInfo	Controller Model Number, Firmware Version, Diagnostic Status, and Application Name	SNVT_str_asc	36	1Not Supported		
	nci*Name*	**Invensys Configuration Properties**						
46	mnc1	nciSatConfig1	Satellite Configuration Property 1 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
47	mnc3	nciSatConfig2	Satellite Configuration Property 2 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
48	mnc4	nciSatConfig3	Satellite Configuration Property 3 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
49	mnc5	nciSatConfig4	Satellite Configuration Property 4 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
50	mnc6	nciSatConfig5	Satellite Configuration Property 5 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
51	mnc7	nciSatConfig6	Satellite Configuration Property 6 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
52	mnc8	nciSatConfig7	Satellite Configuration Property 7 (defined by the application)	SNVT_count_inc_f	52	—	—	AV
53	mnc9	nciSatConfig8	Satellite Configuration Property 8 (defined by the application)	SNVT_count_inc_f	52	—	—	AV

Table–B.8 Invensys LONMARK VAV Profile Items: Version 2 - Program ID 80:00:16:50:0A:04:04:09 and Version 3 - Program ID 80:00:16:50:0A:04:04:0A

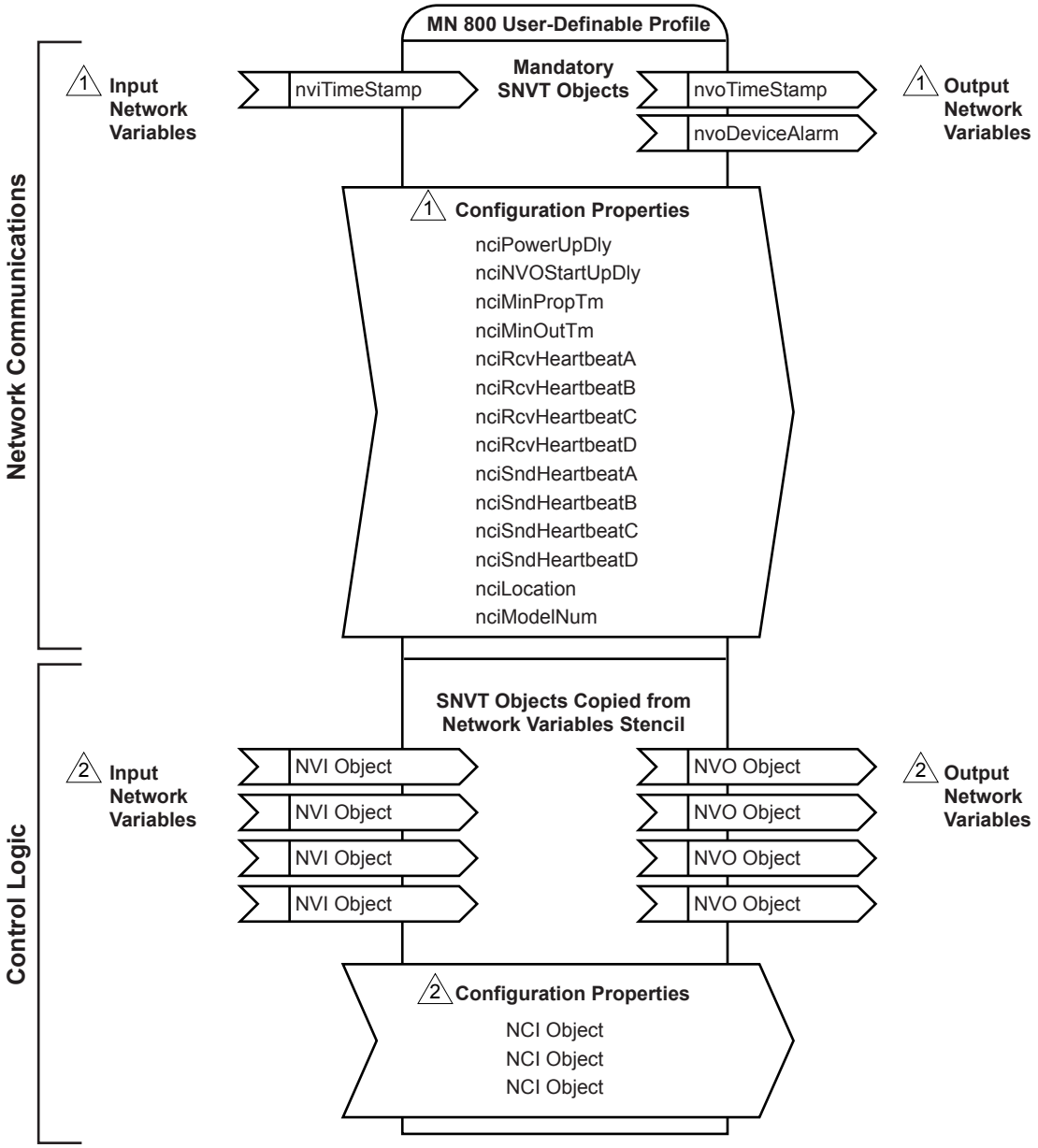
NV Index	Self Doc Index	Profile Item Name	Typical Usage Description	SNVT Name	SNVT Type	Elements	Element Numbers	Point Type
	nci*Name*	**Invensys Configuration Properties (continued)**						
54[b]	mnc2	nciSECModelNum	Controller Part Number and Application Number	SNVT_str_asc	36	1Not Supported		
55	mnc12	nciBalanceOvrd	Provides Air Balance Interface	SNVT_hvac_overid	111	.State .Percent .Flow	1 2 3	AV AV AV
56	mnc13	nciHighFlowCal	Flow Calibration @ 1.0” WC	SNVT_flow	15	—	—	AV
57	mnc14	nciLowFlowCal	Flow Calibration @ 0.1” WC	SNVT_flow	15	—	—	AV
58	mnc15	nciParallelFanSP	Parallel Fan Setpoint	SNVT_flow	15	—	—	AV
59	mnc16	nciRotation	Box Damper Rotation	SNVT_count_inc	9	—	—	AV
60	mnc17	nciFanOvrd	Box Fan Override	SNVT_count_inc	9	—	—	AV

a. In version 3 profile only

b. In version 1 or 2 profile only

MN 800 User-Definable Profile

The MN 800's user-definable profile is graphically shown below Figure-9.



1 Each MN 800 contains a core set of mandatory SNVT objects that are always present in its LON-visible profile. These SNVT objects provide the controller with a basic "network image" that is used to set up the controller characteristics (heartbeats, start up delays, time stamp, and location information). In an MN 800 application in WP Tech, shapes representing these default SNVT objects cannot be deleted (similar to resource tags, such as hardware tags and sensor tags).

2 After configuring the mandatory SNVT objects, the controller's default "network image" may be expanded with additional SNVT objects copied from the Network Variables stencil to the application drawing.

Figure–B.9 MN 800 Controller User-Definable Profile.

MN 800 Profile Quick Reference

Each MN 800 contains a core set of mandatory SNVT objects Table–B.9 that are always present in its LON-visible profile. These SNVT objects define the network communications characteristics of the controller. In an MN 800 application in WP Tech, shapes representing these default SNVT objects cannot be deleted (similar to resource tags, such as hardware tags and sensor tags).

Table–B.9 Mandatory (Default) MN 800 SNVT Objects.

NCI objects		NVI objects	NVO objects
nciLocation	nciRcvHeartbeatC	nviTimeStamp	nvoDeviceAlarm
nciMinOutTm	nciRcvHeartbeatD		nvoTimeStamp
nciMinPropTm	nciModelNum		
nciNVOStartUpDly	nciSndHeartbeatA		
nciPowerUpDly	nciSndHeartbeatB		
nciRcvHeartbeatA	nciSndHeartbeatC		
nciRcvHeartbeatB	nciSndHeartbeatD		

Refer to the “MicroNet MN 800 Controllers” section in Chapter 6 of this manual, beginning on page 548, for detailed information related to the MN 800’s user-definable profile.

SNVTs

Standard Network Variable Types (SNVTs) promote device interoperability by standardizing a wide range of specific data types. Of the 170-plus SNVTs defined in the LONMARK document, *The SNVT Master List and Programmer's Guide*, about 20 types are used among the profiles in I/A Series MicroNet standard controllers.

SNVT to MicroNet Control Logic Data Exchange

All data processed by the "object engine" of a MicroNet standard controller must fall within the controller's numbering system: -163.83 (minimum) to 16,383 (maximum). This applies to all data produced internally and also to any data produced externally, that is, brought inside the application using an NVI or NCI of the controller's LONMARK profile. Note, however, that many SNVTs have a numerical range that exceeds this controller number range. Therefore, it is possible that data outside of this -163.83 to 16,383 range (that is otherwise valid) will enter a MicroNet controller and be used in the control object logic. When this occurs, this data is evaluated as the closest endpoint of this range (as either -163.83 or 16,383). *No over-range error or warning is issued.*

For example, in a VAV controller profile, if nciMaxFlow (using SNVT_flow) is modified in WP Tech, the compiler prevents any value over the upper limit of 16383 (liters/second). However, if this NCI is modified using a third-party network management program, any value up to 65,534 is permitted due to the specified range of SNVT_flow. Nonetheless, any control object input connected to this modified NCI will evaluate the value as 16383.

Invalid data and not active (NA)

Note that for some SNVTs, a particular value is defined as "invalid data". Typically, this value is numerically one over the top range of valid SNVT data. For example, 65535 is invalid data for SNVT_flow (range is from 0 to 65534). For those SNVTs *without* a LONMARK "invalid data" definition, the MicroNet controller treats the highest possible valid value as invalid, or not active (NA).

In a MicroNet controller, invalid data values and not active (NA) values are direct equivalents. This equivalence applies to any profile item in a MicroNet controller, as described below.

NCIs: An NCI with NA assigned from WP Tech contains the SNVT's invalid data value when seen from a third-party network management tool. If an invalid value is not defined, the value will be the highest possible value.

NVIs: An NVI that is bound to an external node and is updated by that node with the SNVT's invalid data value will be seen as NA in the MicroNet controller (at any connected control object or tag input). If the SNVT used has no invalid value definition, an update with the highest valid value will be evaluated in the MicroNet controller as NA.

NVOs: An NVO that is sourced from a control object or tag output producing an NA output will cause the SNVT's invalid data value to be transmitted from the profile to any externally bound node. If the SNVT used has no invalid value definition, an update with the highest valid value will be transmitted from the profile.

Invalid data and not active (NA) values are noted in the following SNVTs used by Invensys in MicroNet standard controllers.

MN Controllers SNVTs Quick Reference

This section contains two lists of SNVTs, Listing by SNVT Type Name (below) and Listing by SNVT Number (page 680). They present all the various types of SNVTs used in implementing the NVIs, NVOs, and NCIs of the eight different Invensys LONMARK profiles and the user-definable profile for an MN 800. Each SNVT shows a measurement variable, value range and resolution, and applicable details for any structured data fields or enumerated values.

Listing by SNVT Type Name

Table–B.10 alphabetically lists the SNVTs by SNVT type name.

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name.

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_amp	Current, A	-3,276.8 to 3,276.7 amps (0.1 A) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	1
SNVT_amp_ac		0 to 65,534 amps (1 A) Fixed point scalar, 2 bytes NA = 65,535 (0xFFFF)	Generic selection in MN 800.	139
SNVT_amp_f		-1E38 to 1E38 amps Floating point, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	48
SNVT_amp_mil	Current, mA	-3,276.8 to 3,276.7 mA (0.1 mA) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	2
SNVT_angle	Phase/Rotation	0 to 65.535 radians (0.001 radians) Fixed point scalar, 2 byte Invensys NA = 65.535 (0xFFFF)	Generic selection in MN 800.	3
SNVT_angle_deg		-359.98 to +360.00 degrees (0.02 degrees). Fixed point scalar, 2 bytes NA = 655.34 (0x7FFF)	Generic selection in MN 800.	104
SNVT_angle_f		-1E38 to 1E38 radians Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	49
SNVT_angle_vel	Angular velocity	-3,276.8 to 3,276.7 radians/sec (0.1 radians/sec) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	4
SNVT_angle_vel_f		-1E38 to 1E38 radians/sec Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	50
SNVT_area	Area, m2 (English: feet2)	0 to 13.1068 m2 (200 mm2) Fixed point scalar, 2 bytes NA = 13.107 (0xFFFF)	Generic selection in MN 800.	110

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_btu_f	Thermal energy	0 to 1E38 BTU Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	67
SNVT_btu_kilo		0 to 65,535 kilo BTU Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	5
SNVT_btu_mega		0 to 65,535 mega BTU Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	6
SNVT_char_ascii	Character	0 to 255 Fixed point scalar, 1 byte Invensys NA = 255 (0xFF)	Generic selection in MN 800.	7

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_chlr_status	Chiller Status	Structured data using these 7 fields: • Run Mode - (chiller_run_mode) - Enumerated value as follows: 0 - (CHLR_OFF) - Chiller off 1 - (CHLR_START) - Chiller in start mode 2 - (CHLR_RUN) - Chiller in run mode 3 - (CHLR_PRESHUTDN) - Chiller in pre-shutdown mode 4 - (CHLR_SERVICE) - Chiller in service mode NA = 255 (0xFF) - (CHLR_NULL) - (also Invensys NA) • Operating Mode - (chlr_op_mode) - Enumerated value as follows: 0 - (HVAC_AUTO) - Controller automatically changes between application modes 1 - (HVAC_HEAT) - Heating only 2 - (HVAC_MRNG_WRMUP) - Application-specific morning warm-up 3 - (HVAC_COOL) - Cooling only 4 - (HVAC_NIGHT_PURGE) - Application-specific night purge 5 - (HVAC_PRE_COOL) - Application-specific pre-cool 6 - (HVAC_OFF) - Controller not controlling outputs 7 - (HVAC_TEST) - Equipment being tested 8 - (HVAC_EMERG_HEAT) - Emergency heat mode (heat pump) 9 - (HVAC_FAN_ONLY) - Air not conditioned, fan turned on 10 - (HVAC_FREE_COOL) - Cooling with compressor not running 11 - (HVAC_ICE) - Ice-making mode NA = 255 (0xFF) - (HVAC_NUL) - (also Invensys NA) • Alarm - (in_alarm), with a value of 0 or 1, where 1 means in alarm • Run Enabled - (run_enabled), with a value of 0 or 1, where 1 means run is enabled • Local - (local), with a value of 0 or 1, where 1 means local • Limited - (limited), with a value of 0 or 1, where 1 means (chiller) limited • Chilled Water Flow - (chw_flow), with a value of 0 or 1, where 1 means chilled water flow • Conditioned Water Flow - (condw_flow), with a value of 0 or 1, where 1 means conditioned water flow	Structured SNVT selection in MN 800.	127
SNVT_config_src	Installation Source	Enumerated value as follows: 0 - (CFG_LOCAL) - Node will use self installation functions to set its own network image 1 - (CFG_EXTERNAL) - Node's network image will be set by an outside source NA = 255 (0xFF) - (also Invensys NA)	Enumerated selection in MN 800.	69

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_count	Event Count	0 to 65,535 counts (1 count) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection	8
SNVT_count_f		0 to 1E38 counts Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	51
SNVT_count_inc	Incremental Count	-32,768 to 32,767 (1 count) Fixed point scalar, 2 bytes Invensys NA = 32,767 (0x7FFF)	Used in VAV profile; nciRotation and nciFanOvrd.	9
SNVT_count_inc_f		-1E38 to 1E38 counts Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Used in all profiles; nciSatConfigx, and other NCIs, NVOs.	52
SNVT_date_day	Day of Week	Enumerated value as follows: 0 - (DAY_SUN) - Sunday 1 - (DAY_MON) - Monday 2 - (DAY_TUE) - Tuesday 3 - (DAY_WED) - Wednesday 4 - (DAY_THU) - Thursday 5 - (DAY_FRI) - Friday 6 - (DAY_SAT) - Saturday NA = 255 (0xFF) - (DAY_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	11
SNVT_defr_mode	Defrost Mode	Enumerated value as follows: 0 - (DFM_MODE_AMBIENT) - No forced heating required 1 - (DFM_MODE_FORCED) - Start up after defrost ignored 2 - (DFM_MODE_SYNC) - Synchronized NA = 255 (0xFF) - (DFM_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	120
SNVT_defr_state	Defrost State	Enumerated value as follows: 0 - (DFS_STANDBY) - Defrost in standby 1 - (DFS_PUMPDOWN) Defrost in pumpdown mode 2 - (DFS_DEFROST) - In defrost mode 3 - (DFS_DRAINDOWN) - Defrost in draindown 4 - (DFS_INJECT_DLY) - Defrost in injection delay NA = 255 (0xFF) - (DFS_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	122
SNVT_defr_term	Defrost Termination	Enumerated value as follows: 0 - (DFT_TERM_TEMP) - Terminate on temperature 1 - (DFT_TERM_TIME) - Terminate on time 2 - (DFT_TERM_FIRST) - Terminate on first occuring 3 - (DFT_TERM_LAST) - Terminate on last occurring NA = 255 (0xFF) - (DFT_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	121

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_density	Density, kg/m3 (English: lb/ft3)	0 to 32,767.5 kg/m3 (0.5 kg/m3) Fixed point scalar, 2 bytes Invensys NA = 32,767.5 (0xFFFF)	Generic selection in MN 800.	100
SNVT_density_f		0 to 1E38 kg/m3 Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	101
SNVT_elapsed_tm	Elapsed Time	Structured data using these 5 fields: • Day (day), with values of 0 to 65,534 NA = 65,535 (0xFFFF) (also Invensys NA) • Hour (hour), with values of 0 to 23 Invensys NA = 0 if day NA • Minute (minute), with values of 0 to 59 Invensys NA = 0 if day NA • Second (second), with values of 0 to 59 Invensys NA = 0 if day NA • Millisecond (millisecond), with values of 0 to 999 Invensys NA = 0 if day NA	Structured SNVT selection in MN 800.	87
SNVT_elec_kwh	Electrical Energy	0 to 65,535 kiloWatt-hour (1 kW-h) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	13
SNVT_elec_whr		0 to 6,553.5 Watt-hours (0.1 W-h) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	14
SNVT_elec_whr_f		0 to 1E38 Watt-hour Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	68
SNVT_evap_state	Evaporation State	Enumerated value as follows: 0 - (EVAP_NO_COOLING) - Object not performing cooling (off cycle or disabled) 1 - (EVAP_COOLING) - Object currently cooling 2 - (EVAP_EMERG_COOLING) - Object performing emergency cooling NA = 255 (0xFF) - (EVAP_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	118
SNVT_flow	Flow, l/s or ml/s (English: CFM)	0 to 65,534 liters/sec (1 l/s) Fixed point scalar, 2 bytes NA = 65,535 (0xFFFF) (also Invensys NA)	Used in VAV profile; various NCIs, NVIs, NVOs.	15
SNVT_flow_f		-1E38 to1E38 liters/sec Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	53
SNVT_flow_mil		0 to 65,535 milliliters/sec (1 ml/sec) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	16

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_freq_f	Frequency	0 to 1E38 Hertz (Hz) Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	75
SNVT_freq_hz		0 to 6553.5 Hz (0.1 Hz) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	76
SNVT_freq_kilohz		0 to 6553.5 kHz (0.1 kHz) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	77
SNVT_freq_milhz		0 to 6.5535 Hz (0.0001 Hz) Fixed point scalar, 2 bytes Invensys NA = 6.5535 (0xFFFF)	Generic selection in MN 800.	78
SNVT_grammage	Grammage	0 to 6,553.5 gsm (0.1 gsm) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	71
SNVT_grammage_f		0 to 1E38 gsm Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	72
SNVT_hvac_emerg	Emergency Mode, HVAC	Enumerated value as follows: 0 - (EMERG_NORMAL) - No emergency mode 1 - (EMERG_PRESSURIZE) - Emergency pressurize mode 2 - (EMERG_DEPRESSURIZE) - Emergency depressurize mode 3 - (EMERG_PURGE) - Emergency purge mode 4 - (EMERG_SHUTDOWN) - Emergency shutdown mode NA = 255 (0xFF) - (EMERG_NUL) - (also Invensys NA)	Used in VAV and RTU profiles; nviEmergCmd. Also an enumerated selection in MN 800.	103
SNVT_hvac_mode	HVAC Mode	Enumerated value as follows: 0 - (HVAC_AUTO) - Controller automatically changes mode. 1 - (HVAC_HEAT) - Heating only 2 - (HVAC_MRNG_WRMUP) - Application-specific morning warmup 3 - (HVAC_COOL) - Cooling only 4 - (HVAC_NIGHT_PURGE) - Application-specific night purge 5 - (HVAC_PRE_COOL) - Application-specific pre-cool 6 - (HVAC_OFF) - Controller not controlling outputs 7 - (HVAC_TEST) - Equipment being tested 8 - (HVAC_EMERG_HEAT) -Emergency heat mode (heat pump) 9 - (HVAC_FAN_ONLY) - Fan only, air not conditioned 10 - (HVAC_FREE_COOL) - Cooling with compressor not running 11 = (HVAC_ICE) - Ice-making mode NA = 255 (0xFF) - (HVAC_NUL) - (also Invensys NA)	Used in all profiles; nviApplicMode or nvoSatApplicMode. Also an enumerated selection in MN 800.	108

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_hvac_overid	HVAC override	Structured data using these 3 fields: • Flow (flow), with flow override value (SNVT_flow) of 0 to 65,534 liters/sec NA = 65,535 (also Invensys NA) • Percent (percent), with percentage position or flow override value (SNVT_lev_percent) of -163.84 to 163.83% (0.005%) NA = 163.835 (also Invensys NA) • State - Enumerated value as follows: 0 - (HVO_OFF) - Not overriden The following 16 override values apply to all devices or groups. 1 - (HVO_POSITION) - Override position percentage (using Percent field value) 2 - (HVO_FLOW_VALUE) - Override Flow in liters/sec (using Flow field value) 3 - (HVO_FLOW_PERCENT) - Override flow percentage (using Percent field value) 4 - (HVO_OPEN) - Override to position = 100% 5 - (HVO_CLOSE) - Override to position = 0% 6 - (HVO_MINIMUM) - Override to configured minimum 7 - (HVO_MAXIMUM) - Override to configured maximum 8 through 16 - (HVO_UNUSED8 through HVO_UNUSED16) The following 16 override values apply to the first device or group 17 - (HVO_POSITION_1) - Override position percentage (using Percent field value) 18 - (HVO_FLOW_VALUE_1) - Override Flow in liters/sec (using Flow field value) 19 - (HVO_FLOW_PERCENT_1) - Override flow percentage (using Percent field value) 20 - (HVO_OPEN_1) - Override to position = 100% 21 - (HVO_CLOSE_1) - Override to position = 0% 22 - (HVO_MINIMUM_1) - Override to configured minimum 23 - (HVO_MAXIMUM_1) - Override to configured maximum 24 through 32 - (HVO_UNUSED24 through HVO_UNUSED32) Continued in next row.	Used in all profiles; nviOverride, nviManOverride, also nciBalanceOvrd in VAV profile. Also a structured SNVT selection in MN 800.	111

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_hvac_overid (cont'd)	HVAC override	The following 16 override values apply to the second device or group 33 - (HVO_POSITION_2) - Override position percentage (using Percent field value) 34 - (HVO_FLOW_VALUE_2) - Override Flow in liters/sec (using Flow field value) 35 - (HVO_FLOW_PERCENT_2) - Override flow percentage (using Percent field value) 36 - (HVO_OPEN_2) - Override to position = 100% 37 - (HVO_CLOSE_2) - Override to position = 0% 38 - (HVO_MINIMUM_2) - Override to configured minimum 39 - (HVO_MAXIMUM_2) - Override to configured maximum 40 through 48 - (HVO_UNUSED40 through HVO_UNUSED48) NA = 255 (0xFF) - (HVO_NUL) - (also Invensys NA)		111

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_hvac_status	HVAC status	Structured data using these 7 fields: • Alarm (in_alarm), with a value of 0 or 1, where 1 means Alarm Invensys NA = 255 (0xFF) Following 5 fields have output values of -163.83 to 163.83% (0.005%) as a percentage of full scale, where NA = 163.835 (0x7FFF) (also Invensys NA): • Cool (cool_output)- Cooling Output • Econ (econ_output) - Economizer Output • Fan - (fan_output) - Fan Output • Heat1 - (heat_output_primary) - Primary Heat Output • Heat2 - (heat_output_secondary) - Secondary Heat Output • Mode (mode) - Enumerated value as follows: 0 - (HVAC_AUTO) - Controller automatically changes mode. 1 - (HVAC_HEAT) - Heating only 2 - (HVAC_MRNG_WRMUP) - Application-specific morning warmup 3 - (HVAC_COOL) - Cooling only 4 - (HVAC_NIGHT_PURGE) - Application-specific night purge 5 - (HVAC_PRE_COOL) - Application-specific pre-cool 6 - (HVAC_OFF) - Not controlling outputs 7 - (HVAC_TEST) - Equipment being tested 8 - (HVAC_EMERG_HEAT) - Emergency heat mode (heat pump) 9 - (HVAC_FAN_ONLY) - Fan only, air not conditioned 10 - (HVAC_FREE_COOL) - Chiller runs with compressor turned off 11 - (HVAC_ICE) - Chiller making ice NA = 255 (0xFF) - (HVAC_NUL) - (also Invensys NA)	Used in all profiles; nvoUnitStatus. Also a structured SNVT selection in MN 800.	112
SNVT_hvac_type	HVAC Unit Type	Enumerated value as follows: 0 - (HVT_GENERIC) - Generic 1 - (HVT_FAN_COIL) - Fan Coil 2 - (HVT_VAV) - Variable Air Volume Terminal 3 - (HVT_HEAT_PUMP) - Heat Pump 4 - (HVT_ROOFTOP) - Rooftop Unit 5 - (HVT_UNIT_VENT) - Unit Ventilator 6 - (HVT_CHILL_CEIL) - Chilled Ceiling 7 - (HVT_RADIATOR) - Radiator 8 - (HVT_AHU) - Air Handling Unit 9 - (HVT_SELF_CONT) - Self-Contained Unit NA = 255 (0xFF) - (HVT_NUL) - (also Invensys NA)	An enumerated selection in MN 800.	145

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_length	Length, meters (English: feet)	0 to 6,553.5 meters (0.1 m) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	17
SNVT_length_f		0 to 1E38 meters (m) Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	54
SNVT_length_kilo	Length, km (English: miles)	0 to 6,533.5 km (0.1 km) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	18
SNVT_length_micr	Length, µm (English: *none*)	0 to 6,553.5 µm (0.1 µm) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	19
SNVT_length_mil	Length, mm (English: *none*)	0 to 6,533.5 mm (0.1 mm) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	20
SNVT_lev_cont	Continuous Level	0 to 100% (0.5%) Fixed point scalar, 1 byte Invensys NA = 255 (0xFF)	Generic selection in MN 800.	21
SNVT_lev_cont_f		0 to 100% Floating point scalar, 1 byte Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	55
SNVT_lev_disc	Discrete Level	Enumerated value as follows: Device Type 2-State 3-State 4-State 0 - (ST_OFF) - Off off off off 1 - (ST_LOW) - Low on low low 2 - (ST_MED) - Medium on high med 3 - (ST_HIGH) - High on high high 4 - (ST_ON) - On on high high NA = 255 (0xFF) null null null - (ST_NUL) - (also Invensys NA)	Used in all profiles; nviSatLevDisc1 and nvoSatLevDisc1. Also an enumerated selection in MN 800.	22
SNVT_lev_percent	Level, percent	-163.84% to 163.83% (0.005% or 50 ppm) Fixed point scalar, 2 bytes NA = 163.835 (0x7FFF) (also Invensys NA)	Used in all profiles; nviSatPercent1, nvoSatPercent1. Generic selection in MN 800.	81
SNVT_lux	Illumination	0 to 65,535 lux (1 lux) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	79
SNVT_mass	Mass, g (English: ounces)	0 to 6,553.5 grams (0.1 g) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	23
SNVT_mass_f		0 to 1E38 grams Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	56
SNVT_mass_kilo	Mass, kg (English: lb)	0 to 6,553.5 kg (0.1 kg) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	24
SNVT_mass_mega	Mass, tonne (English: ton)	0 to 6,553.5 metric tons (0.1 tonne) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	25

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_mass_mil	Mass, mg (English: ounces)	0 to 6,553.5 milligrams (0.1 mg) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	26
SNVT_muldiv	Gain[a]	Structured data using these 2 fields: • Multiplier, with a value of 0 to 65,535, unsigned 16-bit multiplier • Divisor, with a value of 1 to 65,535, unsigned 16-bit divisor Invensys NA = 65,535 (0xFFFF)	Structured SNVT selection in MN 800.	91
SNVT_multiplier	Multiplier	0 to 32.7675 (0.0005) Fixed point scalar, 2 bytes Invensys NA = 32.7675 (0xFFFF)	Generic selection in MN 800.	82
SNVT_occupancy	Occupancy	Enumerated value as follows: 0 - (OC_OCCUPIED) - Area is occupied 1 - (OC_UNOCCUPIED) - Area is unoccupied 2 - (OC_BYPASS) - Area is temporarily occupied for the bypass period 3 - (OC_STANDBY) - Area is temporarily unoccupied (standby) NA = 255 (0xFF) - (OC_NUL) - (also Invensys NA)	Used in all profiles; nviOccCmd, nvoOccCmd. Also an enumerated selection in MN 800.	109
SNVT_override	Override	Enumerated value as follows: 0 - (OV_RETAIN) 1 - (OV_SPECIFIED) 2 - (OV_DEFAULT NA = 255 (0xFF) - (OV_NULL) - (also Invensys NA)	Enumerated selection in MN 800.	97
SNVT_ph	pH	-32.768 to 32.767 pH (0.001 pH) Fixed point scalar, 2 bytes Invensys NA = 32.767 (0x7FFF)	Generic selection in MN 800.	125
SNVT_ph_f		-1E38 to 1E38 pH Fixed point scalar, 4 bytes Invensys NA = 2E38	Generic selection in MN 800.	126
SNVT_power	Power	0 to 6,553.5 Watts (0.1 W) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	27
SNVT_power_f		-1E38 to 1E38 Watts Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	57
SNVT_power_kilo		0 to 6,553.5 kW (0.1 kW) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	28
SNVT_ppm	Concentration	0 to 65,535 parts per million (1 ppm) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF) (also Invensys NA)	Used in most profiles; nviCO2 nviSatPPM, etc. Generic selection in MN 800.	29
SNVT_ppm_f		0 to 1E38 ppm Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	58

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_press	Pressure, Gauge, kPa, Pa (English: inWC)	-3,276.8 to 3,276.7 kiloPascals (0.1 kPa) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	30
SNVT_press_f		-1E38 to 1E38 Pascals Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	59
SNVT_press_p		- 32,768 to + 32,766 Pascals (1 Pa) Fixed point scalar, 2 bytes NA = 32767 (0x7FFF) (also Invensys NA)	Used in VAV profile; nvoVelPressure. Generic selection in MN 800.	113
SNVT_pwr_fact	Power Factor	-1.0 to 1.0 (0.00005) Fixed point scalar, 2 bytes Invensys NA = 1.63835 (0x7FFF)	Generic selection in MN 800.	98
SNVT_pwr_fact_f		-1.0 to 1.0 Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	99
SNVT_res	Resistance	0 to 6,553.5 Ohms (0.1 Ohms) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	31
SNVT_res_f		0 to 1E38 Ohms Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	60
SNVT_res_kilo		0 to 6,553.5 kOhms (0.1 kOhms) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	32
SNVT_rpm	Angular Velocity	0 to 65,534 revolutions/minute (1 RPM). Fixed point scalar, 2 bytes NA = 65,535 (0xFFFF) (also Invensys NA)	Generic selection in MN 800.	102
SNVT_smo_obscur	Smoke Obscuration	0 to 5 % (0.001 %) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	129
SNVT_sound_db	Sound Level	-327.68 to 327.67 decibels (0.01 dB) Fixed point scalar, 2 bytes Invensys NA = 327.67 (0x7FFF)	Generic selection in MN 800.	33
SNVT_sound_db_f		-1E38 to 1E38 dBspl Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	61
SNVT_speed	Speed, m/s (English: ft/sec)	0 to 6,553.5 meters/sec (0.1 m/s) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	34
SNVT_speed_f		-1E38 to 1E38 meters/second Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	62
SNVT_speed_mil		0 to 65.535 meters/second (0.001 m/s) Fixed point scalar, 2 bytes Invensys NA = 65.535 (0xFFFF)	Generic selection in MN 800.	35

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

<table>
<tr><th>SNVT</th><th>Measurement</th><th>Range (Resolution) / Structure / Enumeration</th><th>MicroNet Controller Notes</th><th>SNVT #</th></tr>
<tr><td>SNVT_state</td><td>State</td><td>Structured data using 16 bit-mapped fields:
• State0 (bit 0)
• State1 (bit 1)
• State2 (bit 2)
• State3 (bit 3)
• State4 (bit 4)
• State5 (bit 5)
• State6 (bit 6)
• State7 (bit 7)
• State8 (bit 8)
• State9 (bit 9)
• State10 (bit 10)
• State11 (bit 11)
• State12 (bit 12)
• State13 (bit 13)
• State14 (bit 14)
• State15 (bit 15)

Where each data field represents a boolean value of either 0 or 1, with the following interpretations:

<table><tr><th>0</th><th>1</th></tr><tr><td>OFF</td><td>ON</td></tr><tr><td>Inactive</td><td>Active</td></tr><tr><td>Disabled</td><td>Enabled</td></tr><tr><td>Low</td><td>High</td></tr><tr><td>False</td><td>True</td></tr><tr><td>Normal</td><td>Alarm</td></tr></table>
Note: No defined invalid (or NA) values exist.
In MN controllers, an NA input at an nvoSatState is equivalent to 0 (OFF).</td><td>Used in both the Satellite 1 and Satellite 2 profiles, nviSatState1 and nvoSatState1.

Also a structured SNVT selection in MN 800.</td><td>83</td></tr>
<tr><td>SNVT_str_asc</td><td>ASCII Character string</td><td>8-bit characters, up to 30 characters
(zero-length string with a null terminator)
No NA</td><td>Used in all profiles; nciLocation, nvoSECalarm. Available in NCI object in MN 800.</td><td>36</td></tr>
<tr><td>SNVT_switch</td><td>Switch</td><td>Structured data using 2 fields:
• Value - Intensity as percentage of full scale, 0 to 100% (0.5%)
Invensys NA = 255 (0xFF)
• State - Binary in action, where
0 means OFF
1 means ON
NA = 255 (0xFF) (also Invensys NA)</td><td>Used in all profiles; nciSatSwitchx and various NVIs, NVOs.

Structured SNVT selection in MN 800.</td><td>95</td></tr>
</table>

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_telcom	phone state	Enumerated value as follows: 0 - (TEL_NOTINUSE) - “Null State (U0)” not in use 1 - (TEL_OFFHOOK) - “Call iInitiated (U1)” 2 - (TEL_DIALING) - “Overlap Sending (U2)” 3 - (TEL_DIALCOMP) - “Outgoing Call Proceeding (U3)” 4 - (TEL_RINGBACK) - “Call Delivered (U4)” hearing ringback 5 - (TEL_INCOMING) - “Call Present (U6)” incoming call has not yet started ringing (only on ISDN line) 6 - (TEL_RINGING) - “Call Received (U7)” incoming call when the user has indicated alerting but has not yet answered 7 - (TEL_ANSWERED) - “Connect Request (U8)” user has answered the call and is waiting to be awarded the call 9 - (TEL_TALKING) - “Active (U10)” two parties are exchanging data 10 - (TEL_HANGINGUP) - “Disconnect Request (U11)” user has hung up 11 - (TEL_HUNGUPX) - “Disconnect Indication (U12)” the other side hung up 12 - (TEL_HOLD) - "Suspend Request (U15)" user has requested the network suspend the call 13 - (TEL_UNHOLD) - "Resume Request (U17)" resume a held call (usually go back to TEL_TALKING) 14 - (TEL_RELEASE) - "Release Request (U19)" user has requested the network to release 15 - (TEL_FULLDUP) - "Overlap Receiving (U25)" user has acknowledged the call and is prepared to receive additional information (if any) in overlap mode non-ISDN values 16 - (TEL_BLOCKED) - connection with blocking, (call waiting disabled) 17 - (TEL_CWAIT) - call waiting coming in 18 - (TEL_DESTBUSY) - destination busy 19 - (TEL_NETBUSY) - problem, network 20 - (TEL_ERROR) - problem, non-network NA = 255 (0xFF) - (TEL_NUL) - value not available (also Invensys NA)	Enumerated SNVT selection in MN 800.	38

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_temp	Temperature, °C	- 274 to 6,279.5 °C (0.1 °C) Fixed point scalar, 2 bytes (Extended Range) Invensys NA = 6279.5 (0xFFFF)	Satellite 2 profile, nviSatERTemp and nvoSatERTemp ("extended range"). Generic selection in MN 800.	39
SNVT_temp_f		-273.17 to 1E38 °C Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	63
SNVT_temp_p		- 273.17 to + 327.66 °C (0.01 °C) Fixed point scalar, 2 bytes NA = 327.67 (0x7FFF) (also Invensys NA)	Used in all profiles; nviSpaceTemp, nvoSpaceTemp, numerous others. Generic selection in MN 800.	105
SNVT_temp_ror	Temperature Rate of Change/Rise	-16384.0 to 16383.0 °C/minute (0.5 °C/min) Fixed point scalar, 2 bytes NA = 16383.5 (0x7FFF) (also Invensys NA)	Generic selection in MN 800.	131
SNVT_temp_setpt	Temperature Setpoints, °C	Structured data using these 6 fields: For all fields, values of -273.17 to 327.66 °C (0.01 °C), NA = 327.67 (0x7FFF) (also Invensys NA) • OccCool - (occupied_cool) - Occupied Cooling • OccHeat - (occupied_heat) - Occupied Heating • SBCool - (standby_cool) - Standby Cooling • SBHeat - (standby_heat) - Standby Heating • UnoccCool - (unoccupied_cool) - Unoccupied Cooling • UnoccHeat - (unoccupied_heat) - Unoccupied Heating	Used in all profiles; nciSetPts. Also a structured SNVT selection in MN 800.	106
SNVT_therm_mode	Thermostat Mode	Enumerated value as follows: 0 - (THERM_NO_CONTROL) - Thermostat disabled 1 - (THERM_IN_OUT) - Cut in/out control 2 - (THERM_MODULATING) - Modulating control NA = 255 (0xFF) - (THERM_NUL) - (also Invensys NA)	An enumerated selection in MN 800.	119

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_time_f	Elapsed Time	0 to 1E38 seconds Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	64
SNVT_time_hour		0 to 65,535 hours (1 hour) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	124
SNVT_time_min		0 to 65,535 minutes (1 minute) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	123
SNVT_time_sec		0.0 to 6,553.4 seconds (0.1 sec) Fixed point scalar, 2 bytes NA = 6553.5 (0xFFFF) (also Invensys NA)	Used in all profiles, nciRcvHrtBt, nciSndHrtBt, nciMinOutTm. Most mandatory NCI objects in an MN 800, also a generic selection.	107
SNVT_time_stamp	Time Stamp	Structured data using 6 fields with these values: • Year - 0* to 3,000 (years AD) NA = 65,535 (0xFFFF) (also Invensys NA) • Month - 0* to 12 • Day - 0* to 31 • Hour - 0 to 23 • Minute - 0 to 59 • Second - 0 to 59 * 0 for Year, Month, or Day means not specified.	Used in all profiles plus the MN 800 nviTimeStamp, nvoTimeStamp.	84
SNVT_tod_event	Supervisory Schedule Command	Structured data using 3 fields: • CurrentState - (current_state) - Current scheduled occupancy state Enumerated as follows: 0 - (OC_OCCUPIED) - Area is occupied 1 - (OC_UNOCCUPIED) - Area is unoccupied 2 - (OC_BYPASS) - Area is temporarily occupied for the bypass period 3 - (OC_STANDBY) - Area is temporarily unoccupied (standby) NA = 255 (0xFF) - (OC_NUL) - (also Invensys NA) • NextState - (next_state) - next scheduled occupancy state Enumerated same as CurrentState • TimeValue - (time_to_next_state) - Minutes until next schedule event change, 0 to 65534 minutes (1 min.) Invensys NA = 65,535 (0xFFFF)	Used in all profiles, nviOccSchedule and nvoOccSchedule. Also a structured SNVT selection in MN 800.	128
SNVT_turbidity	Single Fixed Point Number[b]	0.000 to 65.534 NTU (0.001 NTU) Single fixed point, 2 bytes NA = 65.535 (0xFFFF) (also Invensys NA)	Generic selection in MN 800.	143
SNVT_turbidity_f	Single Floating Point Number[c]	0 to 1E38 NTU Single floating point, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	144

Table–B.10 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Name. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_vol	Volume, liters, kiloliters, milliliters (English: ft3)	0 to 6,553.5 liters (0.1 l) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	41
SNVT_vol_f		0 to 1E38 liters Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	65
SNVT_vol_kilo		0 to 6,553.5 kiloliters (0.1 kl) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	42
SNVT_vol_mil		0 to 6,553.5 milliliters (0.1 ml) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	43
SNVT_volt	Voltage	-3,276.8 to 3,276.7 Volts (0.1 V) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	44
SNVT_volt_ac		0 to 65,534 Vac (1 Vac) Fixed point scalar, 2 bytes NA = 65,535 (0xFFFF) (also Invensys NA)	Generic selection in MN 800.	138
SNVT_volt_dbmv		-327.68 to 327.67 dB µV (0.01 dB µV DC) Fixed point scalar, 2 bytes Invensys NA = 327.67 (0x7FFF)	Generic selection in MN 800.	45
SNVT_volt_f		-1E38 to 1E38 Volts Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	66
SNVT_volt_kilo		-3,276.8 to 3,276.7 kiloVolts (0.1 kV) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	46
SNVT_volt_mil		-3,276.8 to 3,276.7 milliVolts (0.1 mV) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	47
SNVT_zerospan	Zero and Span	Structured data using 2 fields: • Zero - (zero offset) - 163.84 to 163.835% (0.005%) Invensys NA = 163.835 (0x7FFF) • Span - (span multiplier) - 0 to 32.7675 (0.0005) Invensys NA = 32.7675 (0xFFFF)	Structured selection in MN 800.	85

a. A configuration network variable of type SNVT_muldiv may be used as a gain factor for fixed point sensor objects. It is designed for use with the Neuron C muldiv() function, which provides a 16 x 16 unsigned multiplication with a 32-bit intermediate result, followed by a 32/16 unsigned division with a 16-bit end result. For more information see the LONMARK document, “Neuron C Reference Guide”.

b. A network variable of type SNVT_turbidity may be used to communicate with water clarity measuring devices (turbidimeters). SNVT_turbidity can be used to represent ranges associated with drinking water. Units of this variable are NTU (Nephelometric Turbidity Units), which are 1-to-1 equivalent to FNU (Formazin Nephelometric Units), and FTU (Formazin Turbidity Units).

c. A network variable of type SNVT_turbidity_f may be used to communicate with water clarity measuring devices (turbidimeters). Units of this variable are NTU (Nephelometric Turbidity Units), which are 1-to-1 equivalent to FNU (Formazin Nephelometric Units), and FTU (Formazin Turbidity Units). SNVT_turbidity_f can be used to represent ranges associated with industrial applications (e.g., typical measurements may be from 0 to 10000 NTU).

Listing by SNVT Number

Table–B.11 numerically lists the SNVTs by SNVT number.

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number.

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_amp	Current, A	-3,276.8 to 3,276.7 amps (0.1 A) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	1
SNVT_amp_mil	Current, mA	-3,276.8 to 3,276.7 mA (0.1 mA) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	2
SNVT_angle	Phase/Rotation	0 to 65.535 radians (0.001 radians) Fixed point scalar, 2 byte Invensys NA = 65.535 (0xFFFF)	Generic selection in MN 800.	3
SNVT_angle_vel	Angular velocity	-3,276.8 to 3,276.7 radians/sec (0.1 radians/sec) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	4
SNVT_btu_kilo	Thermal energy	0 to 65,535 kilo BTU Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	5
SNVT_btu_mega	Thermal energy	0 to 65,535 mega BTU Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	6
SNVT_char_ascii	Character	0 to 255 Fixed point scalar, 1 byte Invensys NA = 255 (0xFF)	Generic selection in MN 800.	7
SNVT_count	Event Count	0 to 65,535 counts (1 count) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection	8
SNVT_count_inc	Incremental Count	-32,768 to 32,767 (1 count) Fixed point scalar, 2 bytes Invensys NA = 32,767 (0x7FFF)	Used in VAV profile; nciRotation and nciFanOvrd.	9
SNVT_date_day	Day of Week	Enumerated value as follows: 0 - (DAY_SUN) - Sunday 1 - (DAY_MON) - Monday 2 - (DAY_TUE) - Tuesday 3 - (DAY_WED) - Wednesday 4 - (DAY_THU) - Thursday 5 - (DAY_FRI) - Friday 6 - (DAY_SAT) - Saturday NA = 255 (0xFF) - (DAY_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	11
SNVT_elec_kwh	Electrical Energy	0 to 65,535 kiloWatt-hour (1 kW-h) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	13
SNVT_elec_whr	Electrical Energy	0 to 6,553.5 Watt-hours (0.1 W-h) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	14
SNVT_flow	Flow, l/s or ml/s (English: CFM)	0 to 65,534 liters/sec (1 l/s) Fixed point scalar, 2 bytes NA = 65,535 (0xFFFF) (also Invensys NA)	Used in VAV profile; various NCIs, NVIs, NVOs.	15

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_flow_mil	Flow, l/s or ml/s (English: CFM)	0 to 65,535 milliliters/sec (1 ml/sec) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	16
SNVT_length	Length, meters (English: feet)	0 to 6,553.5 meters (0.1 m) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	17
SNVT_length_kilo	Length, km (English: miles)	0 to 6,533.5 km (0.1 km) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	18
SNVT_length_micr	Length, µm (English: *none*)	0 to 6,553.5 µm (0.1 µm) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	19
SNVT_length_mil	Length, mm (English: *none*)	0 to 6,533.5 mm (0.1 mm) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	20
SNVT_lev_cont	Continuous Level	0 to 100% (0.5%) Fixed point scalar, 1 byte Invensys NA = 255 (0xFF)	Generic selection in MN 800.	21
SNVT_lev_disc	Discrete Level	Enumerated value as follows: Device Type 2-State 3-State 4-State 0 - (ST_OFF) - Off off off off 1 - (ST_LOW) - Low on low low 2 - (ST_MED) - Medium on high med 3 - (ST_HIGH) - High on high high 4 - (ST_ON) - On on high high NA = 255 (0xFF) null null null - (ST_NUL) - (also Invensys NA)	Used in all profiles; nviSatLevDisc1 and nvoSatLevDisc1. Also an enumerated selection in MN 800.	22
SNVT_mass	Mass, g (English: ounces)	0 to 6,553.5 grams (0.1 g) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	23
SNVT_mass_kilo	Mass, kg (English: lb)	0 to 6,553.5 kg (0.1 kg) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	24
SNVT_mass_mega	Mass, tonne (English: ton)	0 to 6,553.5 metric tons (0.1 tonne) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	25
SNVT_mass_mil	Mass, mg (English: ounces)	0 to 6,553.5 milligrams (0.1 mg) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	26
SNVT_power	Power	0 to 6,553.5 Watts (0.1 W) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	27
SNVT_power_kilo	Power	0 to 6,553.5 kW (0.1 kW) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	28
SNVT_ppm	Concentration	0 to 65,535 parts per million (1 ppm) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF) (also Invensys NA)	Used in most profiles; nviCO2 nviSatPPM, etc. Generic selection in MN 800.	29

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_press	Pressure, Gauge, kPa, Pa (English: inWC)	-3,276.8 to 3,276.7 kiloPascals (0.1 kPa) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	30
SNVT_res	Resistance	0 to 6,553.5 Ohms (0.1 Ohms) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	31
SNVT_res_kilo	Resistance	0 to 6,553.5 kOhms (0.1 kOhms) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	32
SNVT_sound_db	Sound Level	-327.68 to 327.67 decibels (0.01 dB) Fixed point scalar, 2 bytes Invensys NA = 327.67 (0x7FFF)	Generic selection in MN 800.	33
SNVT_speed	Speed, m/s (English: ft/sec)	0 to 6,553.5 meters/sec (0.1 m/s) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	34
SNVT_speed_mil	Speed, m/s (English: ft/sec)	0 to 65.535 meters/second (0.001 m/s) Fixed point scalar, 2 bytes Invensys NA = 65.535 (0xFFFF)	Generic selection in MN 800.	35
SNVT_str_asc	ASCII Character string	8-bit characters, up to 30 characters (zero-length string with a null terminator) No NA	Used in all profiles; nciLocation, nvoSECalarm. Available in NCI object in MN 800.	36

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_telcom	phone state	Enumerated value as follows: 0 - (TEL_NOTINUSE) - “Null State (U0)” not in use 1 - (TEL_OFFHOOK) - “Call iInitiated (U1)” 2 - (TEL_DIALING) - “Overlap Sending (U2)” 3 - (TEL_DIALCOMP) - “Outgoing Call Proceeding (U3)” 4 - (TEL_RINGBACK) - “Call Delivered (U4)” hearing ringback 5 - (TEL_INCOMING) - “Call Present (U6)” incoming call has not yet started ringing (only on ISDN line) 6 - (TEL_RINGING) - “Call Received (U7)” incoming call when the user has indicated alerting but has not yet answered 7 - (TEL_ANSWERED) - “Connect Request (U8)” user has answered the call and is waiting to be awarded the call 9 - (TEL_TALKING) - “Active (U10)” two parties are exchanging data 10 - (TEL_HANGINGUP) - “Disconnect Request (U11)” user has hung up 11 - (TEL_HUNGUPX) - “Disconnect Indication (U12)” the other side hung up 12 - (TEL_HOLD) - "Suspend Request (U15)" user has requested the network suspend the call 13 - (TEL_UNHOLD) - "Resume Request (U17)" resume a held call (usually go back to TEL_TALKING) 14 - (TEL_RELEASE) - "Release Request (U19)" user has requested the network to release 15 - (TEL_FULLDUP) - "Overlap Receiving (U25)" user has acknowledged the call and is prepared to receive additional information (if any) in overlap mode non-ISDN values 16 - (TEL_BLOCKED) - connection with blocking, (call waiting disabled) 17 - (TEL_CWAIT) - call waiting coming in 18 - (TEL_DESTBUSY) - destination busy 19 - (TEL_NETBUSY) - problem, network 20 - (TEL_ERROR) - problem, non-network NA = 255 (0xFF) - (TEL_NUL) - value not available (also Invensys NA)	Enumerated SNVT selection in MN 800.	38
SNVT_temp	Temperature, ˚C	- 274 to 6,279.5 ˚C (0.1 ˚C) Fixed point scalar, 2 bytes (Extended Range) Invensys NA = 6279.5 (0xFFFF)	Satellite 2 profile, nviSatERTemp and nvoSatERTemp (“extended range”). Generic selection in MN 800.	39
SNVT_vol	Volume, liters (English: ft3)	0 to 6,553.5 liters (0.1 l) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	41

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_vol_kilo	Volume, kiloliters (English: ft3)	0 to 6,553.5 kiloliters (0.1 kl) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	42
SNVT_vol_mil	Volume, milliliters (English: ft3)	0 to 6,553.5 milliliters (0.1 ml) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	43
SNVT_volt	Voltage	-3,276.8 to 3,276.7 Volts (0.1 V) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	44
SNVT_volt_dbmv	Voltage	-327.68 to 327.67 dB µV (0.01 dB µV DC) Fixed point scalar, 2 bytes Invensys NA = 327.67 (0x7FFF)	Generic selection in MN 800.	45
SNVT_volt_kilo	Voltage	-3,276.8 to 3,276.7 kiloVolts (0.1 kV) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	46
SNVT_volt_mil	Voltage	-3,276.8 to 3,276.7 milliVolts (0.1 mV) Fixed point scalar, 2 bytes Invensys NA = 3276.7 (0x7FFF)	Generic selection in MN 800.	47
SNVT_amp_f	Current, A	-1E38 to 1E38 amps Floating point, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	48
SNVT_angle_f	Phase/Rotation	-1E38 to 1E38 radians Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	49
SNVT_angle_vel_f	Angular velocity	-1E38 to 1E38 radians/sec Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	50
SNVT_count_f	Event Count	0 to 1E38 counts Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	51
SNVT_count_inc_f	Incremental Count	-1E38 to 1E38 counts Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Used in all profiles; nciSatConfigx, and other NCIs, NVOs.	52
SNVT_flow_f	Flow, l/s or ml/s (English: CFM)	-1E38 to1E38 liters/sec Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	53
SNVT_length_f	Length, meters (English: feet)	0 to 1E38 meters (m) Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	54
SNVT_lev_cont_f	Continuous Level	0 to 100% Floating point scalar, 1 byte Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	55
SNVT_mass_f	Mass, g (English: ounces)	0 to 1E38 grams Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	56
SNVT_power_f	Power	-1E38 to 1E38 Watts Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	57
SNVT_ppm_f	Concentration	0 to 1E38 ppm Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	58

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_press_f	Pressure, Gauge, kPa, Pa (English: inWC)	-1E38 to 1E38 Pascals Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	59
SNVT_res_f	Resistance	0 to 1E38 Ohms Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	60
SNVT_sound_db_f	Sound Level	-1E38 to 1E38 dBspl Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	61
SNVT_speed_f	Speed, m/s (English: ft/sec)	-1E38 to 1E38 meters/second Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	62
SNVT_temp_f	Temperature, ˚C	-273.17 to 1E38 ˚C Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	63
SNVT_time_f	Elapsed Time	0 to 1E38 seconds Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	64
SNVT_vol_f	Volume, liters (English: ft3)	0 to 1E38 liters Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	65
SNVT_volt_f	Voltage	-1E38 to 1E38 Volts Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	66
SNVT_btu_f	Thermal energy	0 to 1E38 BTU Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	67
SNVT_elec_whr_f	Electrical Energy	0 to 1E38 Watt-hour Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	68
SNVT_config_src	Installation Source	Enumerated value as follows: 0 - (CFG_LOCAL) - Node will use self installation functions to set its own network image 1 - (CFG_EXTERNAL) - Node's network image will be set by an outside source NA = 255 (0xFF) - (also Invensys NA)	Enumerated selection in MN 800.	69
SNVT_grammage	Grammage	0 to 6,553.5 gsm (0.1 gsm) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	71
SNVT_grammage_f	Grammage	0 to 1E38 gsm Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	72
SNVT_freq_f	Frequency	0 to 1E38 Hertz (Hz) Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	75
SNVT_freq_hz	Frequency	0 to 6553.5 Hz (0.1 Hz) Fixed point scalar, 2 bytes Invensys NA = 6553.5 (0xFFFF)	Generic selection in MN 800.	76

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

<table>
<tr><th>SNVT</th><th>Measurement</th><th>Range (Resolution) / Structure / Enumeration</th><th>MicroNet Controller Notes</th><th>SNVT #</th></tr>
<tr><td>SNVT_freq_kilohz</td><td>Frequency</td><td>0 to 6553.5 kHz (0.1 kHz)
Fixed point scalar, 2 bytes
Invensys NA = 6553.5 (0xFFFF)</td><td>Generic selection in MN 800.</td><td>77</td></tr>
<tr><td>SNVT_freq_milhz</td><td>Frequency</td><td>0 to 6.5535 Hz (0.0001 Hz)
Fixed point scalar, 2 bytes
Invensys NA = 6.5535 (0xFFFF)</td><td>Generic selection in MN 800.</td><td>78</td></tr>
<tr><td>SNVT_lux</td><td>Illumination</td><td>0 to 65,535 lux (1 lux)
Fixed point scalar, 2 bytes
Invensys NA = 65,535 (0xFFFF)</td><td>Generic selection in MN 800.</td><td>79</td></tr>
<tr><td>SNVT_lev_percent</td><td>Level, percent</td><td>-163.84% to 163.83% (0.005% or 50 ppm)
Fixed point scalar, 2 bytes
NA = 163.835 (0x7FFF) (also Invensys NA)</td><td>Used in all profiles; nviSatPercent1, nvoSatPercent1. Generic selection in MN 800.</td><td>81</td></tr>
<tr><td>SNVT_multiplier</td><td>Multiplier</td><td>0 to 32.7675 (0.0005)
Fixed point scalar, 2 bytes
Invensys NA = 32.7675 (0xFFFF)</td><td>Generic selection in MN 800.</td><td>82</td></tr>
<tr><td>SNVT_state</td><td>State</td><td>Structured data using 16 bit-mapped fields:
• State0 (bit 0)
• State1 (bit 1)
• State2 (bit 2)
• State3 (bit 3)
• State4 (bit 4)
• State5 (bit 5)
• State6 (bit 6)
• State7 (bit 7)
• State8 (bit 8)
• State9 (bit 9)
• State10 (bit 10)
• State11 (bit 11)
• State12 (bit 12)
• State13 (bit 13)
• State14 (bit 14)
• State15 (bit 15)

Where each data field represents a boolean value of either 0 or 1, with the following interpretations:

<table><tr><th>0</th><th>1</th></tr><tr><td>OFF</td><td>ON</td></tr><tr><td>Inactive</td><td>Active</td></tr><tr><td>Disabled</td><td>Enabled</td></tr><tr><td>Low</td><td>High</td></tr><tr><td>False</td><td>True</td></tr><tr><td>Normal</td><td>Alarm</td></tr></table>
Note: No defined invalid (or NA) values exist.
In MN controllers, an NA input at an nvoSatState is equivalent to 0 (OFF).</td><td>Used in both the Satellite 1 and Satellite 2 profiles, nviSatState1 and nvoSatState1.

Also a structured SNVT selection in MN 800.</td><td>83</td></tr>
<tr><td>SNVT_time_stamp</td><td>Time Stamp</td><td>Structured data using 6 fields with these values:
• Year - 0* to 3,000 (years AD)
NA = 65,535 (0xFFFF) (also Invensys NA)
• Month - 0* to 12
• Day - 0* to 31
• Hour - 0 to 23
• Minute - 0 to 59
• Second - 0 to 59

* 0 for Year, Month, or Day means not specified.</td><td>Used in all profiles plus the MN 800 nviTimeStamp, nvoTimeStamp.</td><td>84</td></tr>
</table>

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_zerospan	Zero and Span	Structured data using 2 fields: • Zero - (zero offset) - 163.84 to 163.835% (0.005%) Invensys NA = 163.835 (0x7FFF) • Span - (span multiplier) - 0 to 32.7675 (0.0005) Invensys NA = 32.7675 (0xFFFF)	Structured selection in MN 800.	85
SNVT_elapsed_tm	Elapsed Time	Structured data using these 5 fields: • Day (day), with values of 0 to 65,534 NA = 65,535 (0xFFFF) (also Invensys NA) • Hour (hour), with values of 0 to 23 InvensysInvensys NA = 0 if day NA • Minute (minute), with values of 0 to 59 Invensys NA = 0 if day NA • Second (second), with values of 0 to 59 Invensys NA = 0 if day NA • Millisecond (millisecond), with values of 0 to 999 Invensys NA = 0 if day NA	Structured SNVT selection in MN 800.	87
SNVT_muldiv	Gain[a]	Structured data using these 2 fields: • Multiplier, with a value of 0 to 65,535, unsigned 16-bit multiplier • Divisor, with a value of 1 to 65,535, unsigned 16-bit divisor Invensys NA = 65,535 (0xFFFF)	Structured SNVT selection in MN 800.	91
SNVT_switch	Switch	Structured data using 2 fields: • Value - Intensity as percentage of full scale, 0 to 100% (0.5%) Invensys NA = 255 (0xFF) • State - Binary in action, where 0 means OFF 1 means ON NA = 255 (0xFF) (also Invensys NA)	Used in all profiles; nciSatSwitchx and various NVIs, NVOs. Structured SNVT selection in MN 800.	95
SNVT_override	Override	Enumerated value as follows: 0 - (OV_RETAIN) 1 - (OV_SPECIFIED) 2 - (OV_DEFAULT NA = 255 (0xFF) - (OV_NULL) - (also Invensys NA)	Enumerated selection in MN 800.	97
SNVT_pwr_fact	Power Factor	-1.0 to 1.0 (0.00005) Fixed point scalar, 2 bytes Invensys NA = 1.63835 (0x7FFF)	Generic selection in MN 800.	98
SNVT_pwr_fact_f	Power Factor	-1.0 to 1.0 Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	99
SNVT_density	Density, kg/m3 (English: lb/ft3)	0 to 32,767.5 kg/m3 (0.5 kg/m3) Fixed point scalar, 2 bytes Invensys NA = 32,767.5 (0xFFFF)	Generic selection in MN 800.	100
SNVT_density_f	Density, kg/m3 (English: lb/ft3)	0 to 1E38 kg/m3 Floating point scalar, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	101
SNVT_rpm	Angular Velocity	0 to 65,534 revolutions/minute (1 RPM). Fixed point scalar, 2 bytes NA = 65,535 (0xFFFF) (also Invensys NA)	Generic selection in MN 800.	102

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_hvac_emerg	Emergency Mode, HVAC	Enumerated value as follows: 0 - (EMERG_NORMAL) - No emergency mode 1 - (EMERG_PRESSURIZE) - Emergency pressurize mode 2 - (EMERG_DEPRESSURIZE) - Emergency depressurize mode 3 - (EMERG_PURGE) - Emergency purge mode 4 - (EMERG_SHUTDOWN) - Emergency shutdown mode NA = 255 (0xFF) - (EMERG_NUL) - (also Invensys NA)	Used in VAV and RTU profiles; nviEmergCmd. Also an enumerated selection in MN 800.	103
SNVT_angle_deg	Phase/Rotation	-359.98 to +360.00 degrees (0.02 degrees). Fixed point scalar, 2 bytes NA = 655.34 (0x7FFF)	Generic selection in MN 800.	104
SNVT_temp_p	Temperature, °C	- 273.17 to + 327.66 °C (0.01 °C) Fixed point scalar, 2 bytes NA = 327.67 (0x7FFF) (also Invensys NA)	Used in all profiles; nviSpaceTemp, nvoSpaceTemp, numerous others. Generic selection in MN 800.	105
SNVT_temp_setpt	Temperature Setpoints, °C	Structured data using these 6 fields: For all fields, values of -273.17 to 327.66 °C (0.01 °C), NA = 327.67 (0x7FFF) (also Invensys NA) • OccCool - (occupied_cool) - Occupied Cooling • OccHeat - (occupied_heat) - Occupied Heating • SBCool - (standby_cool) - Standby Cooling • SBHeat - (standby_heat) - Standby Heating • UnoccCool - (unoccupied_cool) - Unoccupied Cooling • UnoccHeat - (unoccupied_heat) - Unoccupied Heating	Used in all profiles; nciSetPts. Also a structured SNVT selection in MN 800.	106
SNVT_time_sec	Elapsed Time	0.0 to 6,553.4 seconds (0.1 sec) Fixed point scalar, 2 bytes NA = 6553.5 (0xFFFF) (also Invensys NA)	Used in all profiles, nciRcvHrtBt, nciSndHrtBt, nciMinOutTm. Most mandatory NCI objects in an MN 800, also a generic selection.	107

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_hvac_mode	HVAC Mode	Enumerated value as follows: 0 - (HVAC_AUTO) - Controller automatically changes mode. 1 - (HVAC_HEAT) - Heating only 2 - (HVAC_MRNG_WRMUP) - Application-specific morning warmup 3 - (HVAC_COOL) - Cooling only 4 - (HVAC_NIGHT_PURGE) - Application-specific night purge 5 - (HVAC_PRE_COOL) - Application-specific pre-cool 6 - (HVAC_OFF) - Controller not controlling outputs 7 - (HVAC_TEST) - Equipment being tested 8 - (HVAC_EMERG_HEAT) -Emergency heat mode (heat pump) 9 - (HVAC_FAN_ONLY) - Fan only, air not conditioned 10 - (HVAC_FREE_COOL) - Cooling with compressor not running 11 = (HVAC_ICE) - Ice-making mode NA = 255 (0xFF) - (HVAC_NUL) - (also Invensys NA)	Used in all profiles; nviApplicMode or nvoSatApplicMode. Also an enumerated selection in MN 800.	108
SNVT_occupancy	Occupancy	Enumerated value as follows: 0 - (OC_OCCUPIED) - Area is occupied 1 - (OC_UNOCCUPIED) - Area is unoccupied 2 - (OC_BYPASS) - Area is temporarily occupied for the bypass period 3 - (OC_STANDBY) - Area is temporarily unoccupied (standby) NA = 255 (0xFF) - (OC_NUL) - (also Invensys NA)	Used in all profiles; nviOccCmd, nvoOccCmd. Also an enumerated selection in MN 800.	109
SNVT_area	Area, m2 (English: feet2)	0 to 13.1068 m2 (200 mm2) Fixed point scalar, 2 bytes NA = 13.107 (0xFFFF)	Generic selection in MN 800.	110

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_hvac_overid	HVAC override	Structured data using these 3 fields: • Flow (flow), with flow override value (SNVT_flow) of 0 to 65,534 liters/sec NA = 65,535 (also Invensys NA) • Percent (percent), with percentage position or flow override value (SNVT_lev_percent) of -163.84 to 163.83% (0.005%) NA = 163.835 (also Invensys NA) • State - Enumerated value as follows: 0 - (HVO_OFF) - Not overriden The following 16 override values apply to all devices or groups. 1 - (HVO_POSITION) - Override position percentage (using Percent field value) 2 - (HVO_FLOW_VALUE) - Override Flow in liters/sec (using Flow field value) 3 - (HVO_FLOW_PERCENT) - Override flow percentage (using Percent field value) 4 - (HVO_OPEN) - Override to position = 100% 5 - (HVO_CLOSE) - Override to position = 0% 6 - (HVO_MINIMUM) - Override to configured minimum 7 - (HVO_MAXIMUM) - Override to configured maximum 8 through 16 - (HVO_UNUSED8 through HVO_UNUSED16) The following 16 override values apply to the first device or group 17 - (HVO_POSITION_1) - Override position percentage (using Percent field value) 18 - (HVO_FLOW_VALUE_1) - Override Flow in liters/sec (using Flow field value) 19 - (HVO_FLOW_PERCENT_1) - Override flow percentage (using Percent field value) 20 - (HVO_OPEN_1) - Override to position = 100% 21 - (HVO_CLOSE_1) - Override to position = 0% 22 - (HVO_MINIMUM_1) - Override to configured minimum 23 - (HVO_MAXIMUM_1) - Override to configured maximum 24 through 32 - (HVO_UNUSED24 through HVO_UNUSED32) Continued in next row.	Used in all profiles; nviOverride, nviManOverride, also nciBalanceOvrd in VAV profile. Also a structured SNVT selection in MN 800.	111

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_hvac_overid (cont'd)	HVAC override	The following 16 override values apply to the second device or group 33 - (HVO_POSITION_2) - Override position percentage (using Percent field value) 34 - (HVO_FLOW_VALUE_2) - Override Flow in liters/sec (using Flow field value) 35 - (HVO_FLOW_PERCENT_2) - Override flow percentage (using Percent field value) 36 - (HVO_OPEN_2) - Override to position = 100% 37 - (HVO_CLOSE_2) - Override to position = 0% 38 - (HVO_MINIMUM_2) - Override to configured minimum 39 - (HVO_MAXIMUM_2) - Override to configured maximum 40 through 48 - (HVO_UNUSED40 through HVO_UNUSED48) NA = 255 (0xFF) - (HVO_NUL) - (also Invensys NA)		111

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_hvac_status	HVAC status	Structured data using these 7 fields: • Alarm (in_alarm), with a value of 0 or 1, where 1 means Alarm Invensys NA = 255 (0xFF) Following 5 fields have output values of -163.83 to 163.83% (0.005%) as a percentage of full scale, where NA = 163.835 (0x7FFF) (also Invensys NA): • Cool (cool_output)- Cooling Output • Econ (econ_output) - Economizer Output • Fan - (fan_output) - Fan Output • Heat1 - (heat_output_primary) - Primary Heat Output • Heat2 - (heat_output_secondary) - Secondary Heat Output • Mode (mode) - Enumerated value as follows: 0 - (HVAC_AUTO) - Controller automatically changes mode. 1 - (HVAC_HEAT) - Heating only 2 - (HVAC_MRNG_WRMUP) - Application-specific morning warmup 3 - (HVAC_COOL) - Cooling only 4 - (HVAC_NIGHT_PURGE) - Application-specific night purge 5 - (HVAC_PRE_COOL) - Application-specific pre-cool 6 - (HVAC_OFF) - Not controlling outputs 7 - (HVAC_TEST) - Equipment being tested 8 - (HVAC_EMERG_HEAT) - Emergency heat mode (heat pump) 9 - (HVAC_FAN_ONLY) - Fan only, air not conditioned 10 - (HVAC_FREE_COOL) - Chiller runs with compressor turned off 11 - (HVAC_ICE) - Chiller making ice NA = 255 (0xFF) - (HVAC_NUL) - (also Invensys NA)	Used in all profiles; nvoUnitStatus. Also a structured SNVT selection in MN 800.	112
SNVT_press_p	Pressure, Gauge, kPa, Pa (English: inWC)	- 32,768 to + 32,766 Pascals (1 Pa) Fixed point scalar, 2 bytes NA = 32767 (0x7FFF) (also Invensys NA)	Used in VAV profile; nvoVelPressure. Generic selection in MN 800.	113
SNVT_evap_state	Evaporation State	Enumerated value as follows: 0 - (EVAP_NO_COOLING) - Object not performing cooling (off cycle or disabled) 1 - (EVAP_COOLING) - Object currently cooling 2 - (EVAP_EMERG_COOLING) - Object performing emergency cooling NA = 255 (0xFF) - (EVAP_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	118

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_therm_mode	Thermostat Mode	Enumerated value as follows: 0 - (THERM_NO_CONTROL) - Thermostat disabled 1 - (THERM_IN_OUT) - Cut in/out control 2 - (THERM_MODULATING) - Modulating control NA = 255 (0xFF) - (THERM_NUL) - (also Invensys NA)	An enumerated selection in MN 800.	119
SNVT_defr_mode	Defrost Mode	Enumerated value as follows: 0 - (DFM_MODE_AMBIENT) - No forced heating required 1 - (DFM_MODE_FORCED) - Start up after defrost ignored 2 - (DFM_MODE_SYNC) - Synchronized NA = 255 (0xFF) - (DFM_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	120
SNVT_defr_term	Defrost Termination	Enumerated value as follows: 0 - (DFT_TERM_TEMP) - Terminate on temperature 1 - (DFT_TERM_TIME) - Terminate on time 2 - (DFT_TERM_FIRST) - Terminate on first occuring 3 - (DFT_TERM_LAST) - Terminate on last occurring NA = 255 (0xFF) - (DFT_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	121
SNVT_defr_state	Defrost State	Enumerated value as follows: 0 - (DFS_STANDBY) - Defrost in standby 1 - (DFS_PUMPDOWN) Defrost in pumpdown mode 2 - (DFS_DEFROST) - In defrost mode 3 - (DFS_DRAINDOWN) - Defrost in draindown 4 - (DFS_INJECT_DLY) - Defrost in injection delay NA = 255 (0xFF) - (DFS_NUL) - (also Invensys NA)	Enumerated selection in MN 800.	122
SNVT_time_min	Elapsed Time	0 to 65,535 minutes (1 minute) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	123
SNVT_time_hour	Elapsed Time	0 to 65,535 hours (1 hour) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	124
SNVT_ph	pH	-32.768 to 32.767 pH (0.001 pH) Fixed point scalar, 2 bytes Invensys NA = 32.767 (0x7FFF)	Generic selection in MN 800.	125
SNVT_ph_f	pH	-1E38 to 1E38 pH Fixed point scalar, 4 bytes Invensys NA = 2E38	Generic selection in MN 800.	126

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_chlr_status	Chiller Status	Structured data using these 7 fields: • Run Mode - (chiller_run_mode) - Enumerated value as follows: 0 - (CHLR_OFF) - Chiller off 1 - (CHLR_START) - Chiller in start mode 2 - (CHLR_RUN) - Chiller in run mode 3 - (CHLR_PRESHUTDN) - Chiller in pre-shutdown mode 4 - (CHLR_SERVICE) - Chiller in service mode NA = 255 (0xFF) - (CHLR_NULL) - (also Invensys NA) • Operating Mode - (chlr_op_mode) - Enumerated value as follows: 0 - (HVAC_AUTO) - Controller automatically changes between application modes 1 - (HVAC_HEAT) - Heating only 2 - (HVAC_MRNG_WRMUP) - Application-specific morning warm-up 3 - (HVAC_COOL) - Cooling only 4 - (HVAC_NIGHT_PURGE) - Application-specific night purge 5 - (HVAC_PRE_COOL) - Application-specific pre-cool 6 - (HVAC_OFF) - Controller not controlling outputs 7 - (HVAC_TEST) - Equipment being tested 8 - (HVAC_EMERG_HEAT) - Emergency heat mode (heat pump) 9 - (HVAC_FAN_ONLY) - Air not conditioned, fan turned on 10 - (HVAC_FREE_COOL) - Cooling with compressor not running 11 - (HVAC_ICE) - Ice-making mode NA = 255 (0xFF) - (HVAC_NUL) - (also Invensys NA) • Alarm - (in_alarm), with a value of 0 or 1, where 1 means in alarm • Run Enabled - (run_enabled), with a value of 0 or 1, where 1 means run is enabled • Local - (local), with a value of 0 or 1, where 1 means local • Limited - (limited), with a value of 0 or 1, where 1 means (chiller) limited • Chilled Water Flow - (chw_flow), with a value of 0 or 1, where 1 means chilled water flow • Conditioned Water Flow - (condw_flow), with a value of 0 or 1, where 1 means conditioned water flow	Structured SNVT selection in MN 800.	127

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_tod_event	Supervisory Schedule Command	Structured data using 3 fields: • CurrentState - (current_state) - Current scheduled occupancy state Enumerated as follows: 0 - (OC_OCCUPIED) - Area is occupied 1 - (OC_UNOCCUPIED) - Area is unoccupied 2 - (OC_BYPASS) - Area is temporarily occupied for the bypass period 3 - (OC_STANDBY) - Area is temporarily unoccupied (standby) NA = 255 (0xFF) - (OC_NUL) - (also Invensys NA) • NextState - (next_state) - next scheduled occupancy state Enumerated same as CurrentState • TimeValue - (time_to_next_state) - Minutes until next schedule event change, 0 to 65534 minutes (1 min.) Invensys NA = 65,535 (0xFFFF)	Used in all profiles, nviOccSchedule and nvoOccSchedule. Also a structured SNVT selection in MN 800.	128
SNVT_smo_obscur	Smoke Obscuration	0 to 5 % (0.001 %) Fixed point scalar, 2 bytes Invensys NA = 65,535 (0xFFFF)	Generic selection in MN 800.	129
SNVT_temp_ror	Temperature Rate of Change/Rise	-16384.0 to 16383.0 °C/minute (0.5 °C/min) Fixed point scalar, 2 bytes NA = 16383.5 (0x7FFF) (also Invensys NA)	Generic selection in MN 800.	131
SNVT_volt_ac	Voltage	0 to 65,534 Vac (1 Vac) Fixed point scalar, 2 bytes NA = 65,535 (0xFFFF) (also Invensys NA)	Generic selection in MN 800.	138
SNVT_amp_ac	Current, A	0 to 65,534 amps (1 A) Fixed point scalar, 2 bytes NA = 65,535 (0xFFFF)	Generic selection in MN 800.	139
SNVT_turbidity	Single Fixed Point Number[b]	0.000 to 65.534 NTU (0.001 NTU) Single fixed point, 2 bytes NA = 65.535 (0xFFFF) (also Invensys NA)	Generic selection in MN 800.	143
SNVT_turbidity_f	Single Floating Point Number[c]	0 to 1E38 NTU Single floating point, 4 bytes Invensys NA = 2E38 (0x7F167699)	Generic selection in MN 800.	144

Table–B.11 SNVTs Used In I/A Series MicroNet Controllers — By SNVT Number. (Continued)

SNVT	Measurement	Range (Resolution) / Structure / Enumeration	MicroNet Controller Notes	SNVT #
SNVT_hvac_type	HVAC Unit Type	Enumerated value as follows: 0 - (HVT_GENERIC) - Generic 1 - (HVT_FAN_COIL) - Fan Coil 2 - (HVT_VAV) - Variable Air Volume Terminal 3 - (HVT_HEAT_PUMP) - Heat Pump 4 - (HVT_ROOFTOP) - Rooftop Unit 5 - (HVT_UNIT_VENT) - Unit Ventilator 6 - (HVT_CHILL_CEIL) - Chilled Ceiling 7 - (HVT_RADIATOR) - Radiator 8 - (HVT_AHU) - Air Handling Unit 9 - (HVT_SELF_CONT) - Self-Contained Unit NA = 255 (0xFF) - (HVT_NUL) - (also Invensys NA)	An enumerated selection in MN 800.	145

a. A configuration network variable of type SNVT_muldiv may be used as a gain factor for fixed point sensor objects. It is designed for use with the Neuron C muldiv() function, which provides a 16 x 16 unsigned multiplication with a 32-bit intermediate result, followed by a 32/16 unsigned division with a 16-bit end result. For more information see the LONMARK document, “Neuron C Reference Guide”.

b. A network variable of type SNVT_turbidity may be used to communicate with water clarity measuring devices (turbidimeters). SNVT_turbidity can be used to represent ranges associated with drinking water. Units of this variable are NTU (Nephelometric Turbidity Units), which are 1-to-1 equivalent to FNU (Formazin Nephelometric Units), and FTU (Formazin Turbidity Units).

c. A network variable of type SNVT_turbidity_f may be used to communicate with water clarity measuring devices (turbidimeters). Units of this variable are NTU (Nephelometric Turbidity Units), which are 1-to-1 equivalent to FNU (Formazin Nephelometric Units), and FTU (Formazin Turbidity Units). SNVT_turbidity_f can be used to represent ranges associated with industrial applications (e.g., typical measurements may be from 0 to 10000 NTU).

Invensys-defined Network Variable Types

Controllers with Version 1 profiles (MNL-10Rx1, -20Rx1, and -VxR1 models) contain two Invensys-defined network variable types in their profiles. One of these, SEC_tod_event, has since been adopted by LONMARK as SNVT_tod_event. The Invensys-defined data types in revision 1 profiles are:

- SEC_alarm.
- SEC_tod_event.

Both data types are structured, as shown below Figure-12.

Table–B.12 Invensys-Defined NVTs Used in Rev 1 Invensys LONMARK Profiles (MNL-10Rx1, -20Rx1, -VxR1 Models).

Invensys-Defined NVT	Measurement	Range (Resolution)	MicroNet Controller Notes	SNVT #
SEC_alarm (structured)	Control Object Alarm, Controller Service Alarm	Structured data using proprietary fields.	Used in all Rev. 1 profiles: nvoSECalarm Proprietary data. Replaced in Rev. 2 profiles for this NVO by SNVT_str_asc.	—

Table–B.12 Invensys-Defined NVTs Used in Rev 1 Invensys LONMARK Profiles (MNL-10Rx1, -20Rx1, -VxR1 Models).

Invensys-Defined NVT	Measurement	Range (Resolution)	MicroNet Controller Notes	SNVT #
SEC_tod_event (structured)	Supervisory Schedule Command	Structured data using 3 fields: • CurrentState - (current_state), • NextState - (next_state), Both enumerated as follows: 0 - (OC_OCCUPIED) - Area is occupied 1 - (OC_UNOCCUPIED) - Area is unoccupied 2 - (OC_BYPASS) - Area is temporarily occupied for the bypass period 3 - (OC_STANDBY) - Area is temporarily unoccupied (standby) NA ½ 255, (OC_NUL) - Occupancy Null • TimeValue - (time_to_next_state) - (minutes until next schedule event change) 0 to 65534 minutes (1 min.) Invalid data = 65535 (same in MN,½ NA).	Used in all Rev.1 profiles: nviOccSchedule and nvoOccSchedule.	— (Equiv. to 128)

Index

A

Abbreviations xxii
Abs Sub/Div object 108–109
Acrobat (PDF) Conventions xxi
Action
  Binary Output object 174
  DUI Expander object 227
  Filter, Analog Input object 134
  Filter, Filter object 258
  Filter, Sensor Input object 472
  Loop Single object 331
  VAV Actuator object 529
Add / Add object 110–111
Add / Div object 112–113
Address (Addr)
  Inputs 92
  Outputs 96
Addressing
  Workstation 42
Alarm objects 46
  Analog Alarm 114
  Binary Alarm 154
Alarm Sequence
  Analog Alarm object 116
  Binary Alarm object 156
  Local Alarm Buffer 118, 157
Analog Alarm object 114–122
  Examples 121
Analog Class
  Inputs 92
  Outputs 97
Analog Input object 123–134
  Filter Constant 134
  Milliamps/Volts 126
  Offset Calibration 133
  Resistance 132
  Thermistor/Balco/Platinum 126
Analog Output object 135–138
  Current Device 137
  Voltage Device 138
Analog Output Priority object 139–144
  Current Device 143
  Priority Inputs 142
  Voltage Device 144
Analog Sequence Mode
  Sequence (10) object 492
  Sequence (3) object 478
  Sequence (6) object 485
Analog to Digital Data 99
AND / AND object 145–148
AND / OR object 149–151
Applications
  Downloading 42
Average object 152–153

B

Balco sensor 126
Binary Alarm object 154–160
  Examples 159
Binary Encoder object 161–164
  Example 164
Binary Input object 165–172
  Direct Configuration 168
  Pulse Configuration 169
  Pulse Constant 171
  Reverse Configuration 169
Binary Output object 173–176
  Direct Configuration 175
  Reverse Configuration 175
Bindings, Network 534, 590

C

Calendar object 177–185
Calibration
  Pressure Transducer 409
Clock functions 77
Clock Tags 30, 80
Clocked SR object 186–188
Communications Adapters 13
Compare 2 object 192–194
Compare object 189–191
  Example 191
Configuration Properties 88–91
  Algorithm-related 91
  Description 89

Name 89
Process Time 90
Connection Wires 33
Constant Data Sources 95
Constant Tag 32, 95, 101
Write to RAM 95
Control Override object 195–197
Examples 197
Controller
profile version numbers 8
Controller Schedule 77
Conversion Units 585–589
Cooling Loop 317
CoolStat tag 62, 72
Count Down object 198–199
Count Up object 200–201
COV Priority object 202–206
Example 205
CurEvent Schedule tag 76
Curve Fit object 207–214
Example 1, Valve 210
Example 2, Cascading 212
Example 3, Sensor 213
Custom Applications 41
Custom objects 82–86
Inputs and Outputs 86
Rules 84

D

Data Types, Mixing 99
Daylight Savings Changeover 77
DayOfWk Schedule tag 76
Deadband
Floating Actuator object 264, 266
Floating Actuator Priority object 275, 277
Demand Meter, Binary Input object 171
Demux Select object 215–217
Derivative Control
Loop Sequenced object 318
Loop Single object 331
Description, Object 89
Digital Class
Inputs 93
Outputs 98
Digital to Analog Data 99
Direct Acting
Thermostat 2 object 524
Thermostat object 520
Display(x) tag 64, 73
Documentation, Applicable IBS xx
Downloading Applications 42
Drift Compensation
Floating Actuator object 265
Floating Actuator Priority object 276
Dual Delay object 218–220
Dual Minimum object 221–223
Example 223
Dual Setpoint Mode 499
DUI Expander object 224–231
Action Configuration 227
Example 231
R2R network required 227
Resistance Combinations 228
Duty Cycle Control
PWM object 425, 426
PWM Priority object 435

E

Economizer Loop 321
Auto Range 322
Controlled 321
Rules 324
EEPROM memory 80
EmerIconStat tag 63, 71
EmerState tag 59, 71
EmerSwitch tag 71
Engineering Process Overview 41
English Conversions 585–589
Enthalpy object 232–242
Enumerated SNVTs 537
Event Data Format 78
Event Indicator object 243–245
Example 245
EXOR object 246–247
External Interface File (XIF)
LonMark Controllers 547
MN 800 552

F

Fan Coil Profile 618–622
Diagram 618
Fan Speed object 248–256
Analog (Variable) Speed 253
Digital (1, 2, or 3-speed) 252
Examples 253
Number of Speeds 251
Used with MN-Sx Sensor, LON 256
Fan(x).State tag 59, 69
Fan(x).Value tag 60, 69
FanStat tag 63, 72
Feedback
Floating Actuator object 264, 265
Floating Actuator Priority object 275, 277
Filter Constant
Analog Input object 134
Filter object 258
Sensor Input object 472
Filter object 257–259
Example 259
Firmware Revisions 54
Firmware revisions 7
Flexible Profile (MN 800) 548, 660–661

Floating Actuator object 260–269
Drive Modes 263
Examples 268
With Feedback 265
Without Feedback 264
Floating Actuator Priority object 270–280
Drive Modes 274
Examples 279
Priority Inputs 274
With Feedback 277
Without Feedback 275
Force Open/Force Close
Floating Actuator object 265, 267, 276, 278
VAV Actuator object 529
Functional objects 46

H

Hardware Tags 32, 55
Heat Pump Profile 623–626
Diagram 623
Heating Loop 319
HeatStat tag 63, 72
High Flow Calibration
Pressure Transducer object 408
VAV Actuator object 529
High Select object 281–282
HVAC Mode(x) tag 60, 70

I

I/O Point objects 44
I/O Points, Controller 54
Initializing Controller Clock 77
Input Classes
Analog 92
Digital 93
Inputs 92
Addr 92
Analog Class 92
Data Sources 93
Digital Class 93
Inverting 101
Prioritized 102
Integral Control
Loop Sequenced object 318
Loop Single object 330
Interlock object 283–287
Example 287
Interlock Sequences 285
Timing Diagrams 286
Interstage Delay (10) object 298–302
Interstage Delay (3) object 288–292
Example 292
Interstage Delay (6) object 293–297
Example 297
Inverting Inputs 101

L

Latch object 303–306
Digital Latch Configuration 305
Sample and Hold Configuration 306
Limit object 307–308
Limit Thermostat object 309–311
Linear Sequence Mode
Sequence (10) object 491
Sequence (3) object 477
Sequence (6) object 483
Linking Objects 33, 94
List of All Objects 105
Local Alarm Buffer 157
Active Alarm 118, 157
Return from Alarm 120, 158
Logic Type objects 47
LonMark Controllers
External Interface File (XIF) 547
Profiles 538–547
LonWorks Network Architecture 534
Loop and Process Control objects 47
Loop Sequenced object 312–327
Economizer Rules 324
Example 326
General Setup Guidelines 325
Loop 1 (Cooling) 317
Loop 2 (Heating) 319
Loop 3 (Economizer) 321
Ramp Start 316, 318, 320, 324
Setpoint Rules 316
Loop Single object 328–333
Example 332
General Setup Guidelines 333
Ramp Start 331
Low Flow Calibration 408
Low Select object 334–335

M

MA Volume object 336–338
Example 338
Manual Summary xxiii
Math Type objects 47
Memory Requirements
By Object 613
Controller 609–613
Controller Capacities 80, 610
Monitoring in WP Tech 82, 612
Metric Conversions 585–589
MicroNet controller
profile versions 8
MicroNet controller revisions 7
MicroNet Controllers
Firmware Revisions 54, 545
Hardware Platforms 2
MN 110 5
MN 50, 100, 250, 200 4, 5, 6
MN VAV 6

Profile Versions 616
MicroNet Sensors 10–12
Milliamps
Analog Input object 127
Analog Output object 137
Analog Output Priority object 143
Minimum Off object 339–341
Minimum On object 342–344
Minimum Out Time Control 547
Mixing Data Types 99
MN 800 Controller 548–552
External Interface File (XIF) 552
Flexible Profile 548, 660–661
SNVT Objects 548
Added from NV stencil 550
Index Number 551
Mandatory (Default) 550
Naming 551
NCI Objects 548
NVI Objects 549
NVO Objects 549
MN-Sx Sensors 10–12
Fan Speed object 256
MN-S5 Emergency Key 71
Override Operation 68
Service Pin Message 68
Momentary Start / Stop object 345–347
Start and Stop Pulse 347
Monitor Tag 32
Mul / Add object 348–349
Mul / Div object 350–351

N

Name, Object 89
Name, SNVT Objects 551
NCI object 553–564
Configuration Properties 556
Outputs 557
NCI Tags 95, 543
Minimum Out Time Control 547
Send Heartbeat Control 547
Structured SNVTs 543
NCIs
NVO-related 546
Network Architecture, LonWorks 534
Network Management Tool 534
Network Variables 534
Number of Speeds
Fan Speed object 251
NV Index, SNVT Objects 551
NVI object 565–573
Configuration Properties 567
Inputs 568
Outputs 569
NVI Tags 544
Receive Heartbeat Control 545
NVIs 94
NVO object 574–584
Configuration Properties 576
Inputs 578, 601
NVO Tags 545
Structured SNVTs 546
NVO-Related NCIs 546
NVs 534
Implementation 537

O

Object Categories 44–48, ??–49
Functional objects 46
I/O Point objects 44
Object Memory Requirements 612, 613
Object Name 89
Object Properties 88–98
Inputs 92
Outputs 96
Object Stencils 27–30, 104
Object Support by Controller 105
Object Tags 31, 32
Object Types
Abs Sub/Div 108–109
Add / Add 110–111
Add / Div 112–113
Analog Alarm 114–122
Analog Input 123–134
Analog Output 135–138
Analog Output Priority 139–144
AND / AND 145–148
AND / OR 149–151
Average 152–153
Binary Alarm 154–160
Binary Encoder 161–164
Binary Input 165–172
Binary Output 173–176
Calendar 177–185
Clocked SR 186–188
Compare 189–191
Compare 2 192–194
Control Override 195–197
Count Down 198–199
Count Up 200–201
COV Priority 202–206
Curve Fit 207–214
Demux Select 215–217
Dual Delay 218–220
Dual Minimum 221–223
DUI Expander 224–231
Enthalpy 232–242
Event Indicator 243–245
EXOR 246–247
Fan Speed 248–256
Filter 257–259
Floating Actuator 260–269
Floating Actuator Priority 270–280

High Select 281–282
Interlock 283–287
Interstage Delay (10) 298–302
Interstage Delay (3) 288–292
Interstage Delay (6) 293–297
Latch 303–306
Limit 307–308
Limit Thermostat 309–311
Loop Sequenced 312–327
Loop Single 328–333
Low Select 334–335
MA Volume 336–338
Minimum Off 339–341
Minimum On 342–344
Momentary Start / Stop 345–347
Mul / Add 348–349
Mul / Div 350–351
NCI 553–564
NVI 565–573
NVO 574–584
Off Delay 352–354
On Delay 355–357
OR / AND 358–360
OR / OR 361–363
OSS 364–405
Pressure Transducer 406–409
Priority Input (2) 410–411
Priority Input (4) 412–414
Priority Value Select 415–417
PWM 418–426
PWM Priority 427–442
Ramp 437–442
Reset 443–448
Schedule 7-Day 449–462
Select 463–465
Sensor Input 466–473
Sequence (10) 488–495
Sequence (3) 474–480
Sequence (6) 481–487
Setpoint Control 496–500
SqRt Mul / Add 501–503
SR Flip-Flop 504–510
Step Driver 507–510
Sub / Add 511–512
Sub / Div 513–514
Sub / Mul 515–516
Sub / Sub 517–518
Thermostat 519–521
Thermostat 2 522–525
VAV Actuator 526–532
Off Delay object 352–354
Offset
Analog Input object 133
Sensor Input object 472
On Delay object 355–357
Online Diagnostics 15
OnStat tag 64, 72
Optional Conversions 585–589
OR / AND object 358–360
OR / OR object 361–363
OSS object 364–405
Output Classes
Analog 97
Digital 98
Outputs 96
Addr 96
Analog Class 97
Digital Class 98
Override
Controller Schedule 79
MN-Sx Sensor 67
OvrdTime tag 65, 67
OvrdTimeRemain tag 61, 67

P

PBOccMode tag 59, 62, 67
Period
PWM object 420
PWM Priority object 429
Platinum sensor 126
Pre-engineered Applications 15, 42
Pressure Transducer object 406–409
Calibration Values 408
Example 409
Prioritized Inputs 102
Priority Input (2) object 410–411
Example 411
Priority Input (4) object 412–414
Example 414
Priority Inputs
Analog Output Priority object 142
Floating Actuator Priority object 274
Priority Type Objects, Compared 417
Priority Value Select object 415–417
Example 417
Process Time 90
Profile Tags 32, 542–547
NCI Tags 95, 543
NVI Tags 544
NVO Tags 545
WP Tech Representations 542
Profile Types 4, 5, 6, 616–661
Fan Coil 618–622
Heat Pump 623–626
Roof Top 627–630
Satellite 1 631–636, 643–648
Satellite 2 637–??, 649–654
VAV 655–659
Profiles, LonMark
Available Types 538
Compliancy 539
Function 538
Representations 539

Properties 88–98
Configuration 88–91
Proportional Control
Loop Sequenced object 317
Loop Single object 330
Pulse Constant 171
Pulse Input 169
Binary Input object 169
Momentary Start / Stop object 347
Purpose of this Manual xix
PWM object 418–426
Fixed Duty Cycle Example 425
Period and Input 420
Time-Proportioned Example 421
PWM Priority object 427–442
Fixed Duty Cycle Example 435
Period and Input 429
Time-Proportioned Example 431

R

R2R network 227
RAM Memory 80
By Controller Model 80
Ramp object 437–442
Standard Analog Ramp 439
Step Change Ramp 441
Ramp Start 316
Loop Single object 331
RcvHrtBt function 545
Receive Heartbeat Control 545
Reference tag 37–40
Reset object 443–448
Direct Reset Example 445
Example 447
Reset Ratio 446
Reverse Reset Example 446
Reset, Controller
Hard Reset 577
Resistance
Analog Input object 132
DUI Expander object 228
Resource Tags 31, 32, 52
Return from Alarm
Analog Alarm object 119
Binary Alarm object 157
Local Alarm Buffer 120, 158
Rev.3
Firmware 54
RcvHrtBt function 545
Resource Tags 54
Reverse Acting
Thermostat 2 object 524
Thermostat object 520
Revisions, firmware 7
Roof Top Profile 627–630
Diagram 627
RoomTemp tag 61, 66
RTD sensor 126

S

Sample and Hold, Latch object 306
Satellite Profiles 616
Satellite 1 631–636, 643–648
Diagram 631
Satellite 2 637–??, 649–654
Diagram 637
Satellite 3
Diagram 643
Satellite 4
Diagram 649
Schedule 7-Day object 449–462
Schedule Tags 32, 75–80
Clock 30, 80
Controller Clock 77
Controller Schedule 77
Event Data Format 78
Schedule Override 79
Select object 463–465
Example 465
Send Heartbeat Control 547
Sensor Input object 466–473
Filter Constant 472
Offset Calibration 472
Sensor Tags 32, 56, 58–74
Sequence (10) object 488–495
Analog Mode 492
Example 495
Linear Mode 491
Sequence (3) object 474–480
Analog Mode 478
Example 480
Linear Mode 477
Sequence (6) object 481–487
Analog Mode 485
Example 487
Linear Mode 483
Sequence Control objects 48
Service Pin Message
Rev.3 Controller Change 697
Sending From MN-Sx Sensor 68
Setpoint Control object 496–500
Dual Setpoint Mode 499
Example 500
Single Setpoint Mode 498
Single vs. Dual Setpoint 498
Setpoint(x) tag 62, 68
Setup Guidelines
Loop Sequenced object 325
Loop Single object 333
Short-cycle Protection 223
Shutdown Sequence
Interlock 285
Simple SNVTs 537
Single Setpoint Mode 498

S-Link Sensor object 58–74
SNVT Objects, MN 800 548
  Index Number 551
  Mandatory (default) objects 550
  Naming 551
  NCI objects 548
  NVI objects 549
  NVO objects 549
  Objects added from NV stencil 550
SNVT Objects, Name 551
SNVT Objects, NV Index 551
SNVTs 534, 662–697
  Characteristics 536
  Definitions 663–697
  Engineering Units 536
  Enumerated 537
  Formats 536
    Enumerated SNVTs 537
    Simple SNVTs 537
    Structured SNVTs 537
  MicroNet Data Exchange 662
  Overview 536
  Simple 537
  Structured 537
  Types 536
SqRt Mul / Add object 501–503
  Example 503
SR Flip-Flop object 504–510
Standard Network Variable Types 534
Start Pulse 347
Statistics, WP Tech Application 82, 612
Status Flags output
  Analog Input object 125
  DUI Expander object 228
  VAV Actuator object 407
Status Output
  Analog Input object 469
  Sensor Input object 473
Stencils 27–30, 104
Step Driver object 507–510
Stop Pulse 347
Structured SNVTs 537
  NCI Tags 543
  NVO Tags 546
Sub / Add object 511–512
Sub / Div object 513–514
Sub / Mul object 515–516
Sub / Sub object 517–518

T

Tag
  variable definition 37
  variable reference 37
Tags 31
Thermistor sensor 126
Thermostat 2 object 522–525
  Direct and Reverse Acting 524
  Example 525
Thermostat object 519–521
  Direct and Reverse Acting 520
  Example 521
Time-Proportioned Control 421, 431
Timer objects 48
TodEvent Schedule tags 76, 78

U

UnoccStat tag 65, 72

V

Variable Data Sources 94
Variable Definition and Reference tags 37–40
  MN 800 39
Variable Definition Tag 32
Variable Reference Tag 32
VAV Actuator object 526–532
  Action 529
  Examples 531
VAV Profile 655–659
  Diagram 655
Volts
  Analog Input object 129
  Analog Output object 138
  Analog Output Priority object 144

W

Wizards
  Workstation Addressing 42
Workstation Addressing
  Wizard 42
WP Tech (WorkPlace Tech Tool) 14
  Engineering Overview 41
  Online Diagnostics 15, 44
  Write to RAM Function 95

X

XIF File
  LonMark Controllers 547
  MN 800 552